Multiscale Methods for Fabrication Design

by

Desai Chen

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2018

Author ....
Department of Electrical Engineering and Computer Science
October 31, 2017

Certified by..
Wojciech Matusik
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Multiscale Methods for Fabrication Design

by

Desai Chen

Submitted to the Department of Electrical Engineering and Computer Science
on October 31, 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

Modern manufacturing technologies such as 3D printing enable the fabrication of objects with extraordinary complexity. Arranging materials to form functional structures can achieve a much wider range of physical properties than in the constituent materials. Many applications have been demonstrated in the fields of mechanics, acoustics, optics, and electromagnetics. Unfortunately, it is difficult to design objects manually in the large combinatorial space of possible designs. Computational design algorithms have been developed to automatically design objects with specified physical properties. However, many types of physical properties are still very challenging to optimize because predictive and efficient simulations are not available for problems such as high-resolution non-linear elasticity or dynamics with friction and impact. For simpler problems such as linear elasticity, where accurate simulation is available, the simulation resolution handled by desktop workstations is still orders of magnitudes below available printing resolutions.

We propose to speed up simulation and inverse design process of fabricable objects by using multiscale methods. Our method computes coarse-scale simulation meshes with data-drive material models. It improves the simulation efficiency while preserving the characteristic deformation and motion of elastic objects. The first step in our method is to construct a library of microstructures with their material properties such as Young's modulus and Poisson's ratio. The range of achievable material properties is called the material property gamut. We developed efficient sampling method to compute the gamut by focusing on finding samples near and outside the currently sampled gamut. Next, with a pre-computed gamut, functional objects can be simulated and designed using microstructures instead of the base materials. This allows us to simulate and optimize complex objects at a much coarser scale to improve simulation efficiency. The speed improvement leads to designs with as many as a trillion voxels to match printer resolutions. It also enables computational design of dynamic properties that can be faithfully reproduced in reality. In addition to efficient design optimization, the gamut representation of the microstructure envelope provides a way to discover templates of microstructures with extremal physical properties. In contrast to work where such templates are constructed by hand, our work enables the first computational method to automatically discovery microstructure templates that arise from voxel representations.

Thesis Supervisor: Wojciech Matusik
Title: Associate Professor of Electrical Engineering and Computer Science

Acknowledgments

Thank you my advisor, Dr. Wojciech Matusik, for your support, innovative ideas and commitment to research on computational fabrication and physics-based simulation.

Thank you my thesis readers, Dr. Justin Solomon and Dr. Markus J. Buehler for providing valuable feedback to this thesis using your expertise.

Thank you Dr. David I. W. Levin for introducing me to the wonderful world of FEM simulation and for providing advise throughout my PhD study.

Thank you Dr. Shinjiro Sueda for your help on data-driven FEM.

Thank you Dr. Danny Kaufman for your knowledge on dynamics simulation and for advocating our project on designing dynamic mechanisms.

Thank you Dr. Bo Zhu and Dr. Mélina Skouras for your knowledge and hard work on topology optimization and fast numerical solvers.

Thank you Dr. Pitchaya Sitthi-Amorn and Dr. Piotr Didyk for your help on designing 3D prints.

Thank you Dr. James Thompkin and Dan Ramirez for your assistance to paper production.

Thank you lab members of the Computational Fabrication Group: Vahid Babaei, Britton Mae Bradley, Tao du, Alexandre Kaspar, Hui Qiao, Adriana Schulz, Maria Shugrina, Andrew Spielberg, and many others for helping me and for being good company.

Thank you Dr. Etienne Vouga, Dr. Uri Ascher and Dr. Eitan Grinspun for insightful comments and suggestions.

Thank you Gaurav Bharaj, Tzu-Mao Li and Lukas Murmann for user study assistance.

Thank you Jiaqi Xing, my dear wife, for your support and understanding throughout the years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Controlling the behavior of a deformable object is difficult— whether it is a biomechanically accurate character model or a high performance 3D printable microstructure. Getting it right requires constant iteration, either performed manually or driven by an automated system. These design iterations often require many expensive physical prototypes that takes months to make and test.

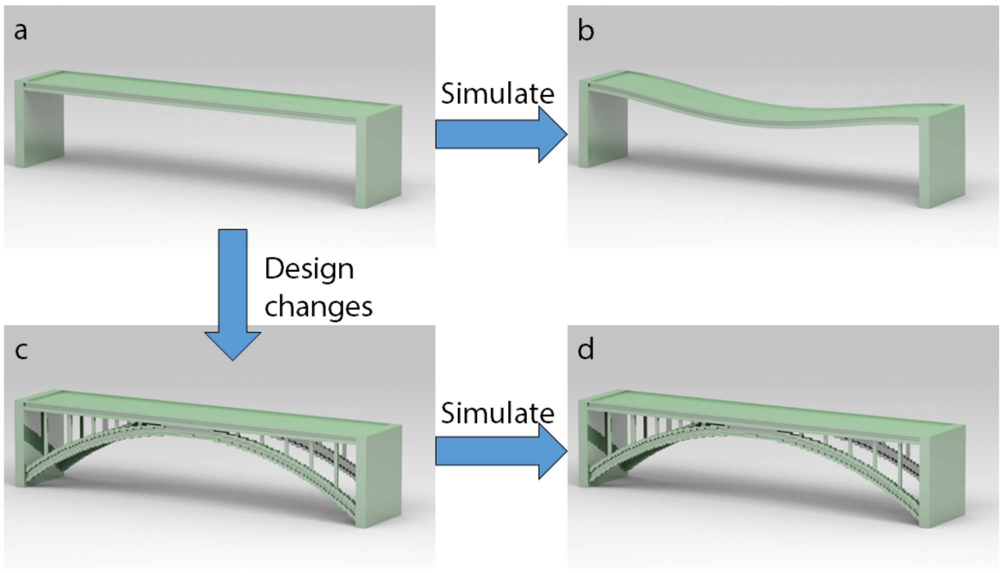


Figure 1-1: Designing a model bridge using simulation tools. The bridge is made of a soft 3D-printing material. (a) Initial design. (b) Simulation predicts that the bridge sags significantly. (c) A designer adds supporting arch to the bridge. (d) The bridge stands without significant deformation.

Researchers and engineers use computational simulation tools to tinker with a virtual design without laboriously constructing new prototypes. The performance of a design is predicted using computation to save time and cost of design iterations (Figure 1-1). For such computational tools to be useful, the simulation algorithm needs to be *predictive* so that simulated results agree with real-world experiments. As a minimal requirement, the simulation should correctly predict whether a significant design modification improves or decreases the physical performance. On top of being predictive, the simulation should be *efficient* to provide interactive feedback to a designer so that she can quickly try different ideas to improve a design.

Taking a step further, a computational design algorithm can automatically tweak design parameters using a predictive simulation algorithm in the loop. Since the simulation often needs to evaluate hundreds or thousands of design variations, the simulation still needs to be fast so that it does not take days to solve a design problem. The gold standard technique for simulating the mechanical behavior of a deformable object is the finite element method (FEM). While FEM is predictive to a high level of accuracy, it is notoriously slow, making it a major computational bottleneck in the iterative design process. The slowness is because FEM is accurate only with a sufficiently high-resolution discretization. To apply FEM to design problems, we require simulation techniques that are fast and accurate even in the face of constant tinkering.

Our goal in this thesis is to develop predictive and efficient FEM simulation tools to enable computational design optimization of deformable objects. The key idea to achieve our goal is *coarsening*. By combining fine elements into a coarse element, the same simulation algorithm uses a much smaller number of degrees of freedom. Since the time complexity for typical simulation algorithms grows superlinearly, coarsening methods can quickly boost the speed of simulation algorithms. The cost associated with coarsening is loss of accuracy. To address this problem, we propose several different techniques to compute new material models for the coarse elements.

To test our simulation in practice, we implemented design optimization algorithms using our simulation for both static and dynamic scenarios. For statics, we modify existing topology optimization algorithms to handle coarse elements that contain microstructures. For dynamic problems, we use a gradient-based optimization algorithm to find locally optimal designs.

Our work draws inspiration from a diverse range of related works in computer graphics and engineering. The next sections provide a brief review of related works on computational design, efficient FEM simulation, and topology optimization. We then conclude this chapter by laying out thesis overview and contributions.

1.1 Computational Design

Computational design concerns itself with optimizing the shape and material assignment in an object in order to control its large scale behavior. Advances in computational design, physical modeling and rapid prototyping have enabled the automated design and fabrication of objects with customized physical properties. The range of fabrication methods and applications addressed in previous literature is very diverse spanning optical properties, inertia, aerodynamic, strength, kinematics, elasticity etc.

In the area of designing optical properties, Hašan et al. [2010] and Dong et al. [2010] provided methods for printing objects with desired subsurface scattering properties. Objects have been made to reflect or bend light to create custom images [Kiser et al., 2013, Papas et al., 2011, Weyrich et al., 2009]. The inertia tensor of an object has been designed for stable standing [Prévost et al., 2013], floating [Musialski et al., 2015] and spinning [Bächer et al., 2014]. The aerodynamic behavior of rigid mechanisms has been controlled to design paper airplanes [Umetani et al., 2014], kites [Martin et al., 2015], and multi-copters [Du et al., 2016]. Many researchers have investigated how to improve structural strength of 3D-printed objects [Langlois et al., 2016, Stava et al., 2012, Ulu et al., 2017, Wu et al., 2016, Zhou et al., 2013]. More complex 3D-printed mechanisms have been designed to achieve specified kinematic motions [Bächer et al., 2015, Coros et al., 2013, Megaro et al., 2017, Zhu et al., 2012].

The most relevant line of work is the design and fabrication of deformable objects with prescribed behavior under load. The behavior is usually controlled by altering the material composition and shape of the designs. Bickel et al. [2010] used a measurement based material model to design layered soft structures with prescribed non-linear static deformation behavior. Chen et al. [2014] optimized the rest shape of a deformable object such that it deforms the right way under gravity. Other researchers have used material composition and geometry to control articulated characters [Bickel et al., 2012, Skouras et al., 2013]. Like these works, one of our goals is to efficiently design compliant objects with desired deformation behavior. These works assume a small set of available base materials such a stiff and a soft material. To expand the range of material building blocks, the base materials are organized into larger structures known as microstructures. These microstructures are then used in design optimization loops to compose objects with specified compliant properties [Panetta et al., 2015, Schumacher et al., 2015, Zhu et al., 2017].

1.2 Efficient FEM Simulation

As mentioned, design and fabrication of deformable objects is common in engineering and graphics [Bendsøe and Sigmund, 2004, Kou et al., 2012, McAdams et al., 2011]. Efficient FEM simulation plays an important role in automating the design process. We can broadly partition the space of approaches for optimizing FEM simulation into two categories. We term the first category numerical approaches. These methods use fast matrix inversion techniques and other insights about the algebra of the FEM to increase its performance. Simulators based on the multigrid method [McAdams et al., 2011, Peraire et al., 1992, Zhu et al., 2010] and Krylov subspace techniques [Patterson et al., 2012] have yielded impressive performance increases. Other hierarchical numerical approaches, as well as highly parallel techniques, have also been applied to improve the time required to perform complex simulations [Farhat and Roux, 1991, Mandel, 1993]. Finally, Bouaziz et al. [2014] have proposed specially designed energy functions and an alternating time-integrator for efficient simulation of dynamics.

The second set of methods are reduction approaches. These algorithms attempt to intelligently decouple or remove degrees of freedom (DOFs) from the simulated system. The reduction leads to smaller systems, resulting in a massive increase in performance, with some decrease in accuracy. Our algorithm falls into this category. Note, however, that numerical and reduction approaches need not be mutually exclusive. For example, since our algorithm uses the same type of spatial discretization as the underlying FEM, faster numerical algorithms such as multigrid will improve the efficiency of our algorithm. Algorithms based on reduction approaches mitigate the inevitable increase in error using one or more of three approaches: adaptive remeshing, higher-order shape functions, or adaption of the constitutive model.

Adaptive remeshing alters the resolution of the simulation discretization in response to various metrics (stress, strain, etc.). Such methods seek to maintain an optimal number of elements and thus achieve reasonable performance. Adaptive remeshing has proven useful for simulating thin sheets such as cloth [Narain et al., 2012], paper [Narain et al., 2013], as well as elastoplastic solids [Wicke et al., 2010] and solid-fluid mixtures [Clausen et al., 2013]. More general basis refinement approaches have also been suggested [Debunne et al., 2001, Grinspun et al., 2002]. While these methods do improve the performance of simulation algorithms, they have some drawbacks. First, they often require complicated geometric operations which can be time consuming to implement. Second, they introduce elements of varying size into the FEM discretization. This can lead to poor numerical conditioning if not done carefully. Finally, in order to maintain accuracy, it may still be necessary to introduce many fine elements, leading to slow performance.

Alternatively, one can turn to P-adaptivity for help. P-adaptivity refers to introducing higher-order basis functions to increase accuracy during simulation [Szabó et al., 2004]. Unfortunately, these methods suffer from requiring complicated mesh generation schemes and are not well-suited for iterative design problems. An alternative approach to remeshing is to use higher order shape functions to more accurately represent the object's motion using a small set of DOFs. Modal simulation techniques fall into this category [Barbič and James, 2005, Krysl et al., 2001, Shabana, 1991]. Substructuring [Barbič and Zhao, 2011] decomposes an input geometry into a collection of basis parts, performing modal reduction on each one. These basis parts can be reused to construct new global structures. Other approaches involve computing physically meaningful shape functions as an offline preprocessing step. For instance, Nesme et al. [2009] compute shape functions based on the static configuration of a high resolution element mesh induced via a small deformation of each vertex. Faure et al. [2011] use skinning transformations as shape functions to simulate complex objects using a small number of frames. Gilles et al. [2011] show how to compute material-aware shape functions for these framebased models, taking into account the linearized object compliance. Both Nesme et al. [2009] and Faure et al. [2011] accurately capture material behavior in the linear regime. However, because their shape functions cannot change with the deformed state of the material, they do not accurately capture the full, non-linear behavior of an elastic object. Our non-linear coarse material models rectify this problem. Computing material-aware shape functions improves both the speed and accuracy of the simulation. However, these methods require a precomputation step that assumes a fixed material distribution and geometry. If the material distribution changes, these shape functions must be recomputed. This step becomes a bottleneck in applications that require constantly changing material parameters.

The final coarsening technique involves reducing the degrees of freedom of a mesh while simultaneously augmenting the constitutive model at each element, rather than the shape functions. Numerical coarsening is an extension of analytical homogenization, which seeks to compute optimal, averaged material for heterogeneous structures [Farmer, 2002, Guedes and Kikuchi, 1990]. Numerical coarsening, for instance, has been applied to linearly (in terms of material displacement) elastic tetrahedral finite elements [Kharevych et al., 2009]. These methods require an expensive precomputation step (a series of static solves) that must be repeated when the material content or the geometry of an object changes. This step holds these methods back from being suitable for iterative design problems.

1.3 Topology Optimization

Topology optimization is a class of methods for optimizing material distributions within a design layout while satisfying given constraints [Bendsøe and Sigmund, 2004]. These methods typically work with linear elastic materials since it is easier to take the derivative of the objective function in this case. Topology optimization was originally developed for structural design problems [Bendsøe, 1989] where the deformation is small enough to be modeled by linear elasticity. It has been extended to a variety of problems including compliant mechanism design [Sigmund, 1997], mass transfer [Challis and Guest, 2009], metamaterial design [Cadman et al., 2013, Sigmund, 2000], multifunctional structure design [Yan et al., 2015], and coupled structure-appearance optimization [Martínez et al., 2015]. Many algorithms have been proposed to numerically solve the optimization problem itself. We refer to the book by Sigmund and Maute [2013] for a complete review.

In the very popular SIMP (Solid Isotropic Materials with Penalization) method, the presence of material in a given cell is controlled by locally varying its density. A binary design is eventually achieved by penalizing intermediate values for these densities. In practice, this method works well for two-material designs (e.g., a material and a void), but generalizing this method to robustly handle higher dimensional material spaces remains challenging. Another class of methods rely on homogenization. They replace the material in each voxel of the object by a mixture of the base materials that is known to be locally optimal for the original problem [Allaire, 2012]. While very powerful, these methods are specialized for the minimum compliance problem, for which the link between stiffness and optimal density can be analytically formulated. In a sense, our work can been seen as a generalization of this type of approaches to handle arbitrary materials in broader contexts.

Standard topology optimization methods suffer from a major drawback. The parametrization of the problem at the voxel level makes them extremely expensive and impedes their use on high resolutions models such as the ones generated by modern 3D printing hardware. To reduce the number of degrees of freedom for simulation and optimization, Rodrigues et al. [2002] proposed an interesting formulation where microstructure designs and macroscopic layouts of the underlying microstructures are hierarchically coupled and treated simultaneously. This initial work has been extended in multiple ways [Coelho et al., 2008, Nakshatrala et al., 2013, Xia and Breitkopf, 2014, Yan et al., 2014]. However, these methods still need to handle variables defined at the microstructure level and therefore they remain relatively costly. The closest to our work is probably the method proposed by Xia and Breitkopf [2015b], which also relies on a database to speed up computations. However, their work specifically targets minimum compliance problem in the structural design which allows them to approximate the macroscale behaviour of the microstructures with a particular strain-based interpolating function.

1.4 Thesis Overview

Chapter 2 introduces FEM for simulating elastic materials. Building on existing material models, we propose data-driven finite elements or DDFEM, the first simulation method optimized for geometric and material design problems under static scenarios. We improve simulation efficiency by coarsening, where we replace blocks of fine elements with coarse elements. We then provide example energy functions for modeling coarse blocks made of heterogeneous Neo-Hookean materials. To quickly find coarse material models for different designs, we use a database to store combinations of fine blocks and their corresponding elastic potential energy functions.

Chapter 3 presents design optimization algorithms based on coarsening. Here the coarse blocks are called microstructures. Microstructures can achieve a much wider range of material properties than the range spanned by the base materials. To automatically explore the achievable material properties of microstructures, we propose to represent the space of material properties as a level set. Based on the level set representation, we develop discrete and continuous sampling methods to explore the space of microstructures more efficiently. We computed the material property space of cubic-symmetric microstructures and discovered families of microstructures that achieves extremal elastic properties. In addition, the level set representation is incorporated into topology optimization algorithms to automatically design objects made of microstructures instead of homogeneous material blocks. This two-scale topology optimization method allows us to design high-resolution objects such as a functional gripping mechanism and a bridge model with maximum stiffness.

Chapter 4 investigates the possibility of designing real-world high-speed dynamic mechanism using coarsened simulations. We propose dynamics-aware coarsening (DAC) to model the dynamic elastic behavior using coarse elements. For impact modeling, we propose boundary-balancing impact (BBI) targeted at inelastic impact of elastic objects. With these methods, we optimize and fabricate jumping mechanisms that are challenging to design by hand.

1.5 Contributions

- DDFEM is the first algorithm for fast runtime coarsening of nonlinear elastic models that can handle changing geometry and material assignment. Given a detailed design, DDFEM coarsens the mesh at interactive rates to yield speed gains of up to two orders of magnitude. The fast coarsening is implemented by precomputing the material properties of coarse elements composed of fine elements with different materials. The material properties are inferred from the deformations of coarse elements under likely forces. A compact material model is developed to represent the coarse element behavior using a small number of parameters.

- Two-scale topology optimization extends existing topology optimization methods to work with microstructures. The extension allows us to optimize structures made of one trillion voxels on a single computer. The software pipeline first precomputes the achievable material properties of microstructures. Instead of optimizing material mixture ratio at each cell, our algorithm optimizes the material properties at each cell while guaranteeing that all material properties are within the achievable range. Our algorithm ensures that the final result is realizable since each cell contains a material property that can be mapped to a microstructure.

- Our efficient microstructure sampling algorithm leads to the computational discovery of microstructure templates with extremal elastic properties. Starting with voxel representation of microstructures, our method automatically cluster similar structures into families and extract parametric templates from the families. The templates are controlled by a small number of parameters to allow more efficient sampling and tweaking. They also reveal the underlying structural similarity that are crucial achieving the extremal physical properties.

- Dynamics-aware coarsening (DAC) computes coarse material properties geared towards dynamic scenarios. DAC achieves a $70x$ simulation speed gain while preserving the overall dynamic motion of designs. The material parameters are computed such that the primary modal frequency of a coarse mesh matches the frequency of a high resolution mesh or a physical model. This allows us to simulate dynamic trajectories of non-linear elastic objects at much faster rates while still matching the macroscopic behavior.

- Boundary-balancing impact (BBI) improves state of the art impact models to more accurately model real-world inelastic impact. We use the combination of DAC and BBI to predict the behavior of elastic mechanisms with loading, contact, friction and impact. For the first time, the simulation of elastic dynamics is predictive and efficient enough to be used in designing physical objects. To show this, we used our simulation in an optimization loop to improve the designs of real-world jumping mechanisms.

Chapter 2

Data-Driven Coarsening of Finite Elements

2.1 Background

2.1.1 The Finite Element Method

The finite element method (FEM) [Ciarlet, 2002] is a numerical method to approximate solutions to partial differential equations. The function domain Ω is divided into a finite number of elements E_k. In most common implementations, a polynomial function is defined over each element. These polynomials form a basis for finite-dimensional approximations to the solution.

This thesis uses the eight-node hexahedron element to perform all of the mechanical analysis. The eight-node hexahedron element is the simplest of the hexahedron family. An element is embedded in $\mathbb{R}^3$ with its eight nodal positions given by $\mathbf{X}_i$. The nodal positions must cooperate to guarantee that the hexahedron is not inverted. In this work, most elements are cubes, and therefore inversion of the rest configuration is avoided by construction. An element represents continuous functions by trilinearly interpolating function values defined on its eight nodes. Each node i is assigned a real value $u(\mathbf{X}_i)$ to define a continuously varying function. For any point $\mathbf{X} \in \mathbb{R}^3$ inside the element, the interpolated value on $\mathbf{X}$ is given by

$$u(\mathbf{X}) = \sum_i N_i(\mathbf{X}) u(\mathbf{X}_i),$$

for some location-dependent weighting functions N_i. These weighting functions are called **shape functions**.

In order to write down the expression for N_i, we need to introduce the standard quadrilateral or hexahedral element (Figure 2-1). The standard hexahedron is just a cube centered at the origin spanning $[-1, 1]^3$. The axis are labeled with ξ, η, ζ to distinguish from the space that $\mathbf{X}$ resides in. This new coordinate system is known as the isoparametric hexahedral coordinates or more commonly referred to as **natural coordinates**. The nodal positions in the natural coordinates are listed in Table 2.1. For a point $\chi = (\xi, \eta, \zeta)$ in the natural

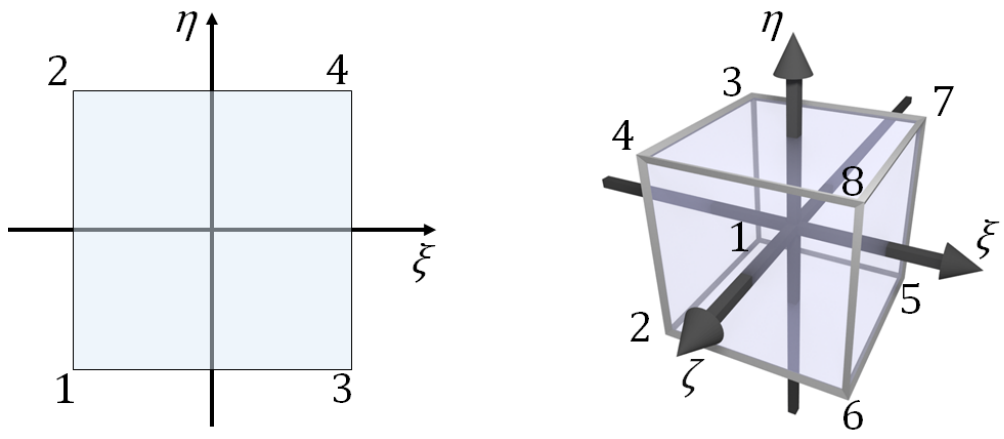


Figure 2-1: Standard elements for 2D quadrilateral (left) and 3D hexahedral elements (right). The standard element spans $[-1, 1]$ in each axis to facilitate Gaussian quadrature rules. Nodes are marked with their indices.

Index	ξ_i	η_i	ζ_i
1	-1	-1	-1
2	-1	-1	1
3	-1	1	-1
4	-1	1	1
5	1	-1	-1
6	1	-1	1
7	1	1	-1
8	1	1	1

Table 2.1: Nodal positions of the standard hexahedron element in the natural coordinates. Coincidentally, these coordinate values can also be used to define the trilinear interpolation weights.

coordinates, its interpolation weights are defined as in Equation 2.1.

$$N_i(\chi) = \frac{1}{8}(1 + \xi_i\xi)(1 + \eta_i\eta)(1 + \zeta_i\zeta). \tag{2.1}$$

Note that the sum of the weights is always 1 for any point inside the element. Suppose now we define a "density" value of 1 on each of its eight vertices, we can compute the total mass of the element by

$$mass = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\sum_i w_i \cdot 1 \, d\xi \, d\eta \, d\zeta = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} 1 \, d\xi \, d\eta \, d\zeta = 8.$$

This is simply the volume of the standard element multiplied by the density. The fact that the eight nodal basis functions form a partition of unity is crucial for physical meanings of quantities defined on the element.

With the interpolation function fully defined in the natural coordinates, we can use it to map a point χ in natural coordinates to a point $\mathbf{X}$ inside a general hexahedron element with nodal positions $\mathbf{X}_i$. Note that the equation interpolates a vector-valued function with each coordinate interpolated independently.

$$\mathbf{X} = \sum_i N_i(\chi)\mathbf{X}_i. \tag{2.2}$$

2.1.2 Modeling Elastic Materials

An object made of elastic materials tend to return to its rest shape when deformed by external forces. The direction and strength of the tendency to restore is described by the elastic forces. To mathematically model the behavior of a small piece of material, we first use finite elements to approximate its deformed shapes. A hexahedron element models a piece of elastic material that can deform by changing its nodal positions. The internal volume deforms by following the nodal displacements using trilinear interpolation. Of course real materials do not have to deform according to piece-wise trilinear interpolation. The interpolation is a way to approximate real deformations using a finite number of degrees of freedom (DOFs). More precisely, for an element with a rest configuration given by $\mathbf{X}_i$, its nodes can be moved to new positions $\mathbf{x}_i$ by displacements $\mathbf{u}_i$, i.e., $\mathbf{x}_i = \mathbf{X}_i + \mathbf{u}_i$. For any point $\mathbf{X}$ inside the element with known natural coordinates χ, its displacement $\mathbf{u}(\mathbf{X})$ is given by interpolation

$$\mathbf{u}(\mathbf{X}) = \sum_i N_i(\chi)\mathbf{u}_i. \tag{2.3}$$

Note the distinction between $\mathbf{X}$ and $\mathbf{x}$. $\mathbf{X}$ is defined as a point in the **reference space** that represents the undeformed shape of an object. The lower case $\mathbf{x}$ lives in the **deformed space** that represents the deformed configuration of an object.

To simplify notation, $N_i(\xi)$ will be written as N_i. The deformation of an object is quantified using strain measures, which is defined in terms of the difference in displacement between nearby points. Intuitively, if $\mathbf{u}(\mathbf{X})$ is constant, then the displacement field is just a translation. In this case, the material is undeformed and should not contain any strain. More generally, the difference of displacement between nearby points can be written using derivatives,

$$\mathbf{F} = \frac{\partial \mathbf{u}}{\partial \mathbf{X}} + \mathbf{I} = \begin{pmatrix} \frac{\partial u_1}{\partial X_1} & \frac{\partial u_1}{\partial X_2} & \frac{\partial u_1}{\partial X_3} \\ \frac{\partial u_2}{\partial X_1} & \frac{\partial u_2}{\partial X_2} & \frac{\partial u_2}{\partial X_3} \\ \frac{\partial u_3}{\partial X_1} & \frac{\partial u_3}{\partial X_2} & \frac{\partial u_3}{\partial X_3} \end{pmatrix} + \mathbf{I}.$$

The matrix $\mathbf{F}$ is the **deformation gradient** and $\mathbf{I}$ is the identity matrix. While we do not have a closed-form expression relating $\mathbf{u}$ and $\mathbf{X}$, we have Equation 2.2 and 2.3 that relate $\mathbf{u}$ and $\mathbf{X}$ through χ. The term $\frac{\partial \mathbf{u}}{\partial \mathbf{X}}$ can be re-written using the chain rule,

$$\frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \frac{\partial \mathbf{u}}{\partial \chi}\frac{\partial \chi}{\partial \mathbf{X}} = (\sum_i \mathbf{u}_i \frac{dN_i}{d\chi})(\frac{\partial \mathbf{X}}{\partial \chi})^{-1} \tag{2.4}$$

The matrix $\frac{\partial \mathbf{X}}{\partial \chi}$ is called the Jacobian matrix of $\mathbf{X}$ with respect to χ and is written as

$$\mathbf{J} = \frac{\partial \mathbf{X}}{\partial \chi} = \sum_i \mathbf{X}_i \frac{dN_i}{d\chi}.$$

By convention, $\frac{dN_i}{d\chi}$ is a 1×3 row vector. The term $\mathbf{X}_i \frac{dN_i}{d\chi}$ is an outer product that produces a 3×3 matrix.

Given the full expression for the deformation gradient $\mathbf{F}$, we can now define elastic energy densities. The elastic energy density function $\Psi(\mathbf{F})$ computes a non-negative energy value given a deformation gradient. This function fully defines the elastic behavior of a piece of material. Integrating Ψ over the domain Ω of an element in the reference space yields the total elastic energy contained in that element.

$$E(\mathbf{x}_1, ..., \mathbf{x}_8) = \int_\Omega \Psi dV.$$

Differentiating E with respect to nodal positions $\mathbf{x}_i$ gives us the opposite direction of nodal elastic forces

$$-\mathbf{f}_i = \frac{dE}{d\mathbf{x}_i} = \int_\Omega \frac{d\Psi(\mathbf{F})}{d\mathbf{F}} \frac{d\mathbf{F}}{d\mathbf{x}_i} dV = \int_\Omega \mathbf{P}(\mathbf{F}) \frac{d\mathbf{F}}{d\mathbf{x}_i} dV,$$

Where

$$\mathbf{P}(\mathbf{F}) = \frac{d\Psi(\mathbf{F})}{d\mathbf{F}}.$$

$\mathbf{P}(\mathbf{F})$ is called the fist Piola-Kirchhoff stress. It transforms a normal vector in the reference space to a force acting in the deformed configuration divided by area in the reference space. Integrating in the reference space is not easy since the element is not necessarily a rectangular shape. By change of variables, we can integrate in natural coordinates in the set $\Omega_\chi = [-1, 1]^3$.

$$\begin{aligned}\int_\Omega \mathbf{P}(\mathbf{F}) \frac{d\mathbf{F}}{d\mathbf{x}_i} dV &= \int_{\Omega_\chi} \mathbf{P}(\mathbf{F}) \frac{d\mathbf{F}}{d\mathbf{x}_i} |\det \mathbf{J}| dV \\ &= \int_{\Omega_\chi} \mathbf{P}(\mathbf{F}) \mathbf{J}^{-T} \frac{dN_i}{d\chi} \det \mathbf{J}\, dV.\end{aligned}$$

Here we dropped the absolute value sign because the undeformed element is required to have positive determinant everywhere. To numerically evaluate the integral, we use Gaussian quadrature rules. For example, the two-point Gaussian quadrature rule places quadrature points at $\pm\sqrt{\frac{1}{3}}$ in natural coordinates, which results in 8 quadrature points with equal weights in 3D. For each quadrature point j, we evaluate the integrand and multiply by the quadrature weight w_j. The integral is written as a summation

$$-\mathbf{f}_i = \sum_j w_j \mathbf{P}(\mathbf{F}_\mathbf{j}) \mathbf{J}^{-T} \frac{dN_i}{d\chi} \det \mathbf{J}.$$

2.1.3 Constitutive Models

In the above section, we defined the strain energy density function Ψ. We also showed how to use it to compute elastic energy and nodal forces without providing concrete formulas for Ψ. The choice of Ψ determines the constitutive model of elasticity, i.e. the relationship between strain measures and stress. Our work primarily uses two kinds of constitutive models: linear elasticity and Neo-Hookean model. For linear elasticity, we first define the infinitesimal strain tensor

$$\epsilon = \frac{1}{2}(\mathbf{F} + \mathbf{F}^T) - I.$$

This strain tensor is a symmetric matrix. The stress tensor derived from this strain measure is also a symmetric matrix. Symmetry of the stress tensor is a necessary condition for conservation of linear and angular momentum. In other words, stress and internal elastic forces should not cause any change in total linear or angular momentum. The strain energy density function for isotropic linear elasticity is

$$\Psi(\mathbf{F}) = \mu\epsilon : \epsilon + \frac{\lambda}{2} tr^2(\epsilon).$$

μ is a material parameter called shear modulus and λ is Lamé's first parameter. $tr(\epsilon)$ is the trace of the strain tensor. Differentiate Ψ to obtain

$$\mathbf{P}(\mathbf{F}) = \mu(\mathbf{F} + \mathbf{F}^T - 2\mathbf{I}) + \lambda tr(\mathbf{F} - \mathbf{I})\mathbf{I}.$$

For anisotropic linear elasticity, the constitutive equation is written as

$$\sigma = \mathbf{C}\epsilon, \tag{2.5}$$

where σ is the 3×3 Cauchy stress tensor, $\mathbf{C}$ is a fourth-order $3 \times 3 \times 3 \times 3$ tensor called the tensor of elasticity. The term $\mathbf{C}\epsilon$ is a summation of component-wise products. In Einstein notation,

$$\sigma_{ij} = C_{ijkl}\epsilon_{kl}.$$

Cauchy stress relates to the first Piola-Kirchhoff stress by

$$\sigma = \frac{1}{\det \mathbf{F}} \mathbf{P}\mathbf{F}^T$$

The corresponding strain energy density function is

$$\Psi(\epsilon) = \frac{1}{2}\mathbf{C}\epsilon^2.$$

To simplify notations, the stress and strain tensors are written as vectors instead of matrices. Using Voigt notation, a stress tensor of the form

$$\sigma = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{pmatrix}$$

is rewritten as

$$\sigma = (\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx}).$$

This allows the tensor of elasticity to be written as a 6×6 matrix. For orthotropic linear materials, $\mathbf{C}$ takes the following form

$$\mathbf{C} = \begin{pmatrix} C_{1111} & C_{1122} & C_{1133} & 0 & 0 & 0 \\ & C_{2222} & C_{2233} & 0 & 0 & 0 \\ & & C_{3333} & 0 & 0 & 0 \\ & & & C_{1212} & 0 & 0 \\ & Symm & & & C_{2323} & 0 \\ & & & & & C_{1313} \end{pmatrix}.$$

For a more restricted subset of materials with cubic symmetry, $\mathbf{C}$ is specified by three material parameters: Young's modulus E, Poisson's ratio ν, and shear modulus G or μ.

$$\mathbf{C} = \begin{pmatrix} (1-\nu)\hat{E} & \nu\hat{E} & \nu\hat{E} & 0 & 0 & 0 \\ & (1-\nu)\hat{E} & \nu\hat{E} & 0 & 0 & 0 \\ & & (1-\nu)\hat{E} & 0 & 0 & 0 \\ & & & \mu & 0 & 0 \\ & Symm & & & \mu & 0 \\ & & & & & \mu \end{pmatrix}, \hat{E} = \frac{E}{(1-2\nu)(1+\nu)}.$$

The advantage of the linear elasticity is that it assumes a linear relationship between stress and strain. This leads to a linear relationship between nodal displacements and nodal forces as follows

$$\mathbf{KU} = -\mathbf{f}.$$

Here $\mathbf{K}$ is called the stiffness matrix given by

$$\mathbf{K} = \frac{\partial^2 \Psi}{\partial \mathbf{U}^2}.$$

$\mathbf{U}$ is the concatenated nodal displacement vector $[\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_8]$ and $\mathbf{f}$ is the concatenated internal nodal force vector. Computing the deformation under external boundary conditions and forces requires only a single linear solve

$$\mathbf{U} = \mathbf{K}^{-1}\mathbf{f}_{ext},$$

given sufficient boundary conditions. The stiffness matrix is sparse and positive semi-definite (positive definite with enough constraints). A wide class of linear solvers have been developed for such linear systems. To see the linear relationship between displacements and force, we just need to derive the linear relationship between ϵ and $\mathbf{u}_i$ from Equation 2.4 and combine it with Equation 2.5. For a point $\mathbf{X} \in \mathbb{R}^3$, we will use superscripts to denote its individual coordinates. Define the strain-displacement matrix

$$\mathbf{B}(\mathbf{X}) = \begin{pmatrix} \frac{dN_1}{d\mathbf{X}^1} & 0 & 0 & \dots \\ 0 & \frac{dN_1}{d\mathbf{X}^2} & 0 & \dots \\ 0 & 0 & \frac{dN_1}{d\mathbf{X}^3} & \dots \\ \frac{dN_1}{d\mathbf{X}^2} & \frac{dN_1}{d\mathbf{X}^1} & 0 & \dots \\ 0 & \frac{dN_1}{d\mathbf{X}^3} & \frac{dN_1}{d\mathbf{X}^2} & \dots \\ \frac{dN_1}{d\mathbf{X}^3} & 0 & \frac{dN_1}{d\mathbf{X}^1} & \dots \end{pmatrix}.$$

For an eight-node hexahedron, this matrix has $3 \times 8 = 24$ columns. This matrix computes the strain tensor at a point $\mathbf{X}$ given nodal displacements

$$\epsilon(\mathbf{X}) = \mathbf{B}(\mathbf{X})\mathbf{U}.$$

This linear relationship concludes our claim that linear elasticity model defines a linear relationship between nodal displacements and nodal forces. Further, using quadrature rules, we can numerically evaluate $\mathbf{K}$ as

$$\mathbf{K} = \sum_j w_j \mathbf{B}(\mathbf{X}_j)\mathbf{C}\mathbf{B}(\mathbf{X}_j) \det \mathbf{J}.$$

The downside of linear elasticity is that it does not handle rotation properly. An elastic object undergoing a rigid rotation should contain zero elastic energy. If we let $\mathbf{F} = \mathbf{R}$ for some rotation matrix $\mathbf{R}$, the resulting energy measure is generally non-zero. Because of this limitation, linear elasticity is useful only when the deformation is small with respect to the overall size of the object. When an object undergoes large deformations such as buckling, it often admits many potential static equilibrium configurations, linear elasticity always predict a single solution with little buckling since its elastic energy landscape is convex with a unique local minimum. Additionally, linear elasticity does not preserve volume according to the Poisson's ratio parameter. In fact, elements can easily invert into non-physical states.

To address the rotation problem, non-linear elasticity models such as the corotated linear elasticity model and the Saint-Venant Kirchhoff model are developed to be rotation invariant. However, these models still allow an element to invert and do not preserve volume. We use the Neo-Hookean material model for non-linear elasticity. It resists inverting, and approximately preserves volume according to Poisson's ratio. The strain energy density of Neo-Hookean model is

$$\Psi(\mathbf{F}) = \frac{\mu}{2}(I_1 - 3) - \mu \log \det \mathbf{F} + \frac{\lambda}{2}(\log \det \mathbf{F})^2,$$

where the first isotropic invariant is

$$I_1 = tr(\mathbf{F}^T\mathbf{F}).$$

The first Piola-Kirchhoff stress is

$$\mathbf{P}(\mathbf{F}) = \mu(\mathbf{F} - \mathbf{F}^{-T}) + \lambda(\log\det\mathbf{F})\mathbf{F}^{-T}.$$

2.2 Data-Driven Coarsening Overview

Objects with high-resolution, heterogeneous elastic materials are everywhere: from the output of multimaterial 3D printers to virtual characters gracing the screen in our summer blockbusters. Designing such objects is made possible by the tight coupling of design tools and numerical simulation which allows designers or automatic algorithms to update geometry or material parameters and subsequently estimate the physical effects of the change. Fast, accurate simulation techniques that can handle runtime changes in geometry and material composition are a necessity for such iterative design algorithms. There have been a large number of works on speeding up FEM simulations, and these speed improvements have enabled FEM to be used in many performance critical tasks such as computer animation, surgical training, and virtual/augmented reality. Even though techniques such as model reduction or numerical coarsening can achieve order-of-magnitude performance increases, they require expensive precomputation phases, typically on the order of minutes for large meshes. This precomputation requires knowledge of an object's geometry and material composition a priori, something that is not known during a design task. When the user updates the model by changing the geometry or the material distribution, the preprocessing step must be run again. As shown in Figure 2-2a, since this step is inside the design loop, the user cannot get rapid feedback on the changes made to the object.

We propose Data-Driven FEM (DDFEM), a new simulation methodology that removes these limitations and is thus extremely well suited to the types of design problems discussed above. We divide an object into a set of deformable voxels using embedded finite elements and coarsen these voxels hierarchically. A custom coarse element database is populated with materials that minimize the error incurred by coarsening. This database is learned once in a completely offline fashion and depends only on the set of materials to be used by the deformable object and not on the actual material distribution and geometry. At runtime we use the database to perform fast coarsening of an FEM mesh in a way that is agnostic to changes in geometry and material composition of the object. The key features of the algorithm are its ability to handle arbitrary, nonlinear elastic constitutive models as well as to avoid expensive precomputation within the design loop (Figure 2-2b). DDFEM is the first algorithm optimized for interactive design of non-linear elastic objects.

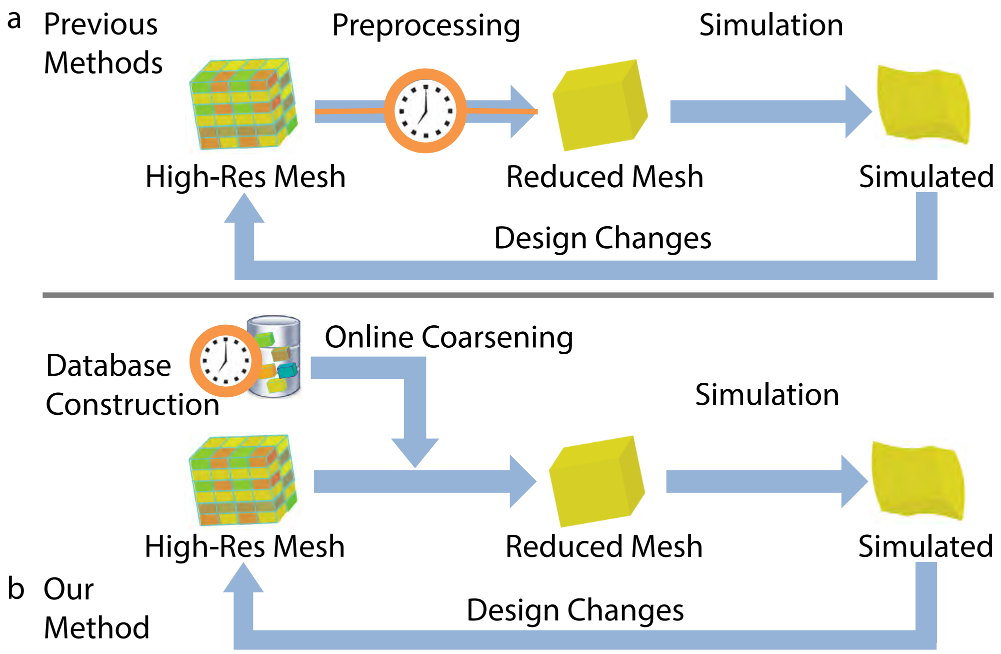


Figure 2-2: (a) In a typical method, the preprocessing step is offline, making the design loop slow. (b) In our method, we move the timeconsuming offline computation outside of the design loop.

DDFEM is a combination of embedded finite elements and hierarchical coarsening. In the remainder of this section, we discuss the problem of coarsening and introduce the notion of a material palette. We conclude by summarizing the two main stages of DDFEMâĂŤoffline coarse material construction and online coarsening.

Coarsening for finite elements The key component of our DDFEM is coarsening. Coarsening reduces the number of vertices in a finite element simulation mesh in order to improve runtime performance. Since simply removing vertices can greatly reduce the accuracy of the simulation, coarsening schemes also assign new materials to coarsened elements to minimize this effect. We regard the global coarsening of a simulation mesh as the result of many local coarsening operations which map from contiguous subsets of fine elements with applied materials to coarse elements with new, optimized materials. Our goal is to precompute these optimized materials so that coarsening is fast at runtime. Below we discuss how to make such a precomputation tractable beginning with our choice of finite element simulation methodology.

Conforming vs. embedded finite elements The defining feature of conforming finite element methods is that the simulation mesh is aligned with the geometry of the object being simulated. One obvious feature of conforming meshes is that the mesh itself is a function of the input geometry. This means that the output of a local coarsening operator (the coarsened mesh) will also be a function of the input geometry. Also, the new material computed by each local coarsening operator will be a function of input geometry. This dependence on input geometry is a significant issue to overcome if we wish to precompute coarsened materials because, in design problems, the input geometry is in constant flux. The number of precomputed coarse materials now depends on the local material assignment on the simulation mesh and the input geometry. Thus space of coarsened materials is prohibitively large. To mitigate this we turn to embedded finite elements. These methods embed the geometry to be simulated into the simulation mesh with no regard for whether the mesh conforms to the geometry or not. Thus an identical simulation mesh can be used for any input geometry. Local coarsening operations on the embedded mesh yield identical coarse elements and the optimized coarse material depends only on the local material distribution on the simulation mesh. This significantly reduces the size of the coarsened material space. In this paper we embed all simulation geometry into a hexahedral simulation mesh.

Material palette We further shrink the space of coarsening operators using an observation about material design. Designers do not work in a continuous space of materials but limit themselves to a relatively compact set (e.g. rubber, wood, steel) related to their problem domain. We call these discrete sets of materials palettes and denote them $\mathcal{P} = \{\mathcal{M}^1, ..., \mathcal{M}^n\}$. Here $\mathcal{M}^i$ denotes a specific material model in $\mathcal{P}$, and n is the size of the material palette. In this work we limit ourselves to nonlinear hyper-elastic materials, which means that each $\mathcal{M}^i$ can be represented by a strain energy density function. We also include a void (or empty) material in every palette. This allows us to perform topology changes in the same manner in which we perform material assignment updates.

Algorithms With the material palette in hand, we can now define our algorithm, which is divided into two distinct phases: an **offline database construction** stage and an **online coarsening** stage. Below we detail the input, output, and steps of each stage:

Offline Database Construction

```
• INPUT: A palette of materials to be applied to high-resolution hexahedral simulation
  meshes $\mathcal{P}^0$
• OUTPUT: A new palette of coarse elements, $\mathcal{P}^1$, and a mapping from fine material
  combinations to the coarsened materials in $\mathcal{P}^1$.
• STEPS:
• FOR EACH material combination applied to a 2×2×2 cube of high resolution ele-
  ments
      • Sample potential energy function of 2×2×2 block
      • Fit coarse hexahedral element material parameters
      • Add coarse element to $\mathcal{P}^1$ using high resolution
      material IDs as database key
• END
```

Online Coarsening

```
• INPUT: High resolution hexahedral simulation mesh with
      material IDs and
      coarsened hexahedral simulation mesh
• OUTPUT: Material assignments for coarse mesh
• STEPS:
• FOR EACH 2×2×2 block in the high resolution mesh
      • Replace with single coarse element
      • Assign material from $\mathcal{P}^1$ using high resolution
      material IDs as database key
• END
```

Hierarchical coarsening We stress that both stages of the DDFEM algorithm can be applied hierarchically. Given the first level of coarse materials, $\mathcal{P}^1$, we can construct a material library, $\mathcal{P}^2$, for the second level by using $\mathcal{P}^1$ as an input material palette. At runtime, the coarsening algorithm looks up materials from $\mathcal{P}^2$ to replace each 2×2×2 coarse block with a single element.

Having introduced the broad strokes of the DDFEM scheme, we move on to a detailed explanation of each algorithmic component. First we discuss database construction in section 2.3, followed by the runtime component in section 2.4. We end by demonstrating the speed and accuracy of DDFEM in section 2.5.

2.3 Coarse Material Database Construction

We construct our coarse material database using a potential energy fitting approach. This is valid due to the hyperelastic materials that make up our material palettes. Material fitting considers 2×2×2 blocks of high-resolution hexahedral elements (denoted $\mathcal{E}^0$). For each element $E^0_k \in \mathcal{E}^0$, its material is referred to as $\mathcal{M}^0_k \in \mathcal{P}^0$. Note that $\mathcal{E}$ refers to a set of elements and E refers to a single element. Given $\mathcal{E}^0$, we can sample its deformation space, and using $\mathcal{M}^0_k$, compute the potential energy V^0 for each sample. Now we must find a coarse material model that, when applied to a single coarse element E^1 best approximates V^0. This is accomplished by fitting a coarse potential energy function , V^1, to the set of deformation/energy samples. The fitted energy is stored in the coarse material database and indexed by the material indices of $\mathcal{M}^0$.

2.3.1 Coarse Material Model

Our fitting approach depends on choosing a good coarse material model. The general hyper-elastic material model for a finite element with degrees of freedom $\mathbf{u}$ can be represented as an energy function $V(\boldsymbol{u}, \boldsymbol{p})$ parameterized by $\boldsymbol{p}$. For an eight-node hexahedron element, $\boldsymbol{u}$ is a 24×1 column vector of nodal displacements $\boldsymbol{u} = (u_{1,x}, u_{1,y}, u_{1,z}..., u_{8,x}, u_{8,y}, u_{8,z})^T$. The energy function has to comply with many requirements to be physically meaningful Marsden and Hughes [2012]. Here we list some important requirements:

1. Invariance under rigid transformation. Given any block-diagonal rotation matrix $\boldsymbol{R}$ composed of 3×3 identical rotation matrices, and any translation vector $\boldsymbol{d}$,
$$V(\boldsymbol{u}, \boldsymbol{p}) = V((\boldsymbol{R} - \boldsymbol{I})\boldsymbol{X} + \boldsymbol{R}\boldsymbol{u} + \boldsymbol{d}, \boldsymbol{p}).$$
2. Tendency to return to the rest shape. To implement this feature, the energy function must have a global minimum of $V(\boldsymbol{u}, \boldsymbol{p}) = 0$ at $\boldsymbol{u} = \boldsymbol{0}$ up to rigid transformations.
3. Stress increases with strain. This is not always true for heterogeneous materials with internal buckling under load such as foams (polymer plus void). For our experiments however, we assume the coarse elements do not undergo deformation large enough to cause internal buckling.
4. Conservation of momentum. The elastic forces given by $\dfrac{DV}{D\boldsymbol{u}}$ must not alter the linear and angular momentum of the element.

These seemingly intuitive requirements are difficult to satisfy for functions written in terms of $\boldsymbol{u}$. For example, here is a simple energy function
$$V(\boldsymbol{u}, \boldsymbol{p}) = \|\boldsymbol{u} - \boldsymbol{X}\|^2.$$
This function violates rules 1,3 and 4. To improve, one can define an energy function by attaching an elastic spring between every pair of nodes. This energy function satisfies rules 1 and 4 but violates rule 2 and 3. To see this, let $\boldsymbol{u} = -2\boldsymbol{X}$ and check that the function has the same value for an inverted element.

In order to ensure that our model meets the criteria for a valid energy function, we choose our coarse material model for E^1 as a combination of the base material models $\mathcal{M}^0_k$:
$$V^1(\boldsymbol{u}^1, \boldsymbol{p}^1) = \sum_{k=1}^{8} w_k \, V^0_k(\boldsymbol{u}^0_k, \boldsymbol{p}^1_k, \boldsymbol{X}^1_k), \tag{2.6}$$

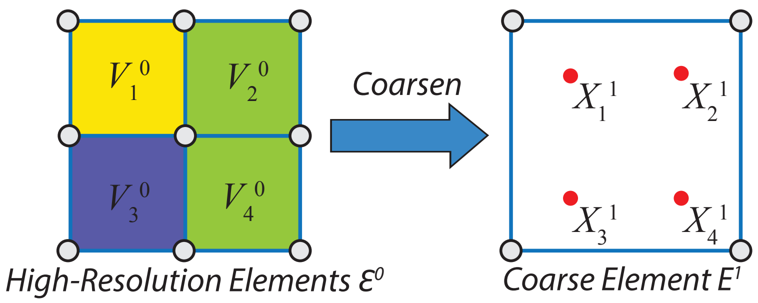


Figure 2-3: The relationship between high-resolution and coarsened elements. At each quadrature point $\boldsymbol{X}_k^1$, the coarse element copies the corresponding energy density function V_k^0 from the high-resolution element.

where V_k^0 is the strain energy density of $\mathcal{M}_k^0$ at quadrature point position $\boldsymbol{X}_k^1$ (Figure 2-3). Here $\boldsymbol{u}^1$ is the vector of nodal displacements associated with E^1 while $\boldsymbol{u}_k^0$ are displacements for the k^{th} element at level 0 reconstructed using trilinear interpolation from $\boldsymbol{u}^1$. The coarse materials $\boldsymbol{p}^1 = (\boldsymbol{p}_1^1, ..., \boldsymbol{p}_k^1)$ consists of the stacked material parameter vectors for each base material in $\mathcal{M}_k^0$, themselves denoted by $\boldsymbol{p}_k^1$. w_k is the standard Gaussian quadrature weight. We observe that even if the individual base material models are isotropic, the coarse element can become anisotropic by assigning different material parameters at the quadrature points. We propose to improve the fitting by augmenting the coarse material model with an anisotropic term. The complete model is then given by

$$\begin{aligned} V^1(\boldsymbol{u}^1, \boldsymbol{p}^1, C) = \sum_{k=1}^{8} \Bigg(& w_k V_k^0(\boldsymbol{u}^0, \boldsymbol{p}_k^1, \boldsymbol{X}_k^1) \\ & + C_k \left(\sqrt{\boldsymbol{v}^T \boldsymbol{F}_k^T \boldsymbol{F}_k \boldsymbol{v}} - 1 \right)^2 \Bigg), \end{aligned} \tag{2.7}$$

where $\boldsymbol{v}$ is a unit-length direction of anisotropy and C_k is the scaling parameter at the k^{th} quadrature point. The gradient of the anisotropic term is

$$\frac{d(\sqrt{\boldsymbol{v}^T \boldsymbol{F}^T \boldsymbol{F} \boldsymbol{v}} - 1)^2}{d\mathbf{F}} = (1 - \frac{1}{\|\mathbf{F}\mathbf{v}\|})\mathbf{F}\mathbf{v}\mathbf{v}^T.$$

The second order gradient in the direction of a given $\delta\mathbf{F}$ is

$$\left((1 - \frac{1}{\|\mathbf{F}\mathbf{v}\|})\mathbf{I} + \frac{1}{\|\mathbf{F}\mathbf{v}\|^3} \boldsymbol{F}\boldsymbol{v}\boldsymbol{v}^T\boldsymbol{F}^T \right) \delta\mathbf{F}^T\mathbf{v}\mathbf{v}^T.$$

2.3.2 Force Space Sampling

As mentioned previously, we take a sampling-based approach to coarse material fitting. In order to fit our model (Equation 2.7) to V^0 we first draw a number of samples from the deformation space of $\mathcal{E}^0$ and compute V^0 for each sample. If a user has prior knowledge

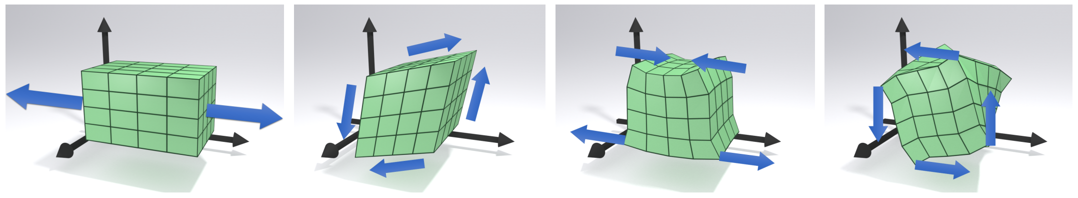


Figure 2-4: Example of forces used in our force-based sampling method. The forces here are stretching, shearing, bending and twisting.

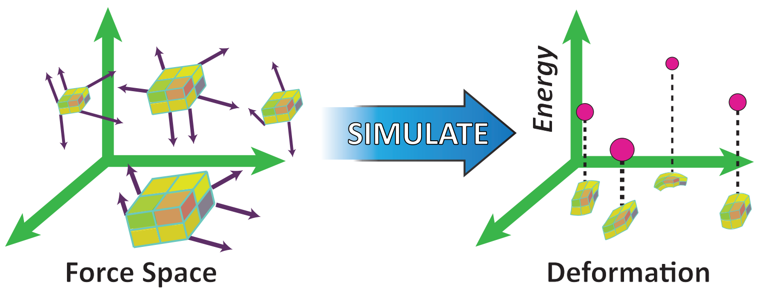


Figure 2-5: We sample the energy function of a 2×2×2 block of hexahedra by applying forces and deforming the block using FEM. Each set of forces results in a deformation-energy tuple which is used during fitting.

of the set of meshes and simulations that they will require, then the best way to draw the samples is to run a number of anticipated simulations with various material combinations. In this paper, we provide a more general method to draw samples for a material model. Initially, we attempted sampling by applying a random deformation to the corners of E^1; however, this led to many infeasible samples for very stiff materials. In order to alleviate this problem we perform sampling in the force space.

For each element $E^0 \in \mathcal{E}^0$ we apply a set of pre-defined forces including stretching, shearing, bending, and twisting(Figure 2-4). Each force is sampled at 5 magnitudes in the positive and negative directions. In addition to the above four kinds of forces, we added a pinching force along each of the twelve cube edges. We solve an elastostatic problem to compute the deformation of $\mathcal{E}^0$, using constraints to remove rigid motion. Recall that this is fast because ${}^0\mathcal{E}$ consists of just 8 elements. Each sample is then a tuple $\{\boldsymbol{u}^0, V^0\}$ (Figure 2-5) where ${}^0\boldsymbol{u}$ are the nodal displacements of $\mathcal{E}^0$, and V^0 is the strain energy density value of this deformed configuration.

2.3.3 Fitting

Given a set of deformation samples, $\{\boldsymbol{u}^0, V^0\}$, we perform a non-negative least squares fit to determine the parameters, $\boldsymbol{p}^*$, for material model(Figure 2-6):

$$\boldsymbol{p}^* = \underset{\boldsymbol{p}^1}{\operatorname{argmin}} \sum_{s=1}^{n_s} \left(V_s^0 - V^1\left(\boldsymbol{r}\left(\boldsymbol{u}_s^0\right), \boldsymbol{p}^1\right)\right)^2, \tag{2.8}$$

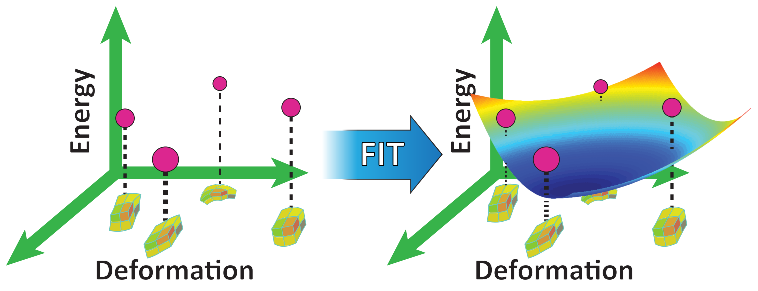


Figure 2-6: Coarse potential energy functions are fitted to the deformation-energy samples using a least-squares optimization.

where $\boldsymbol{r}$ constructs $\boldsymbol{u}^1$ from $\boldsymbol{u}^0$, n_s is the total number of samples, and s indexes all samples. In our experiments we use the simplest form of $\boldsymbol{r}$ choosing it to extract the displacements of the corners of $\mathcal{E}^0$. The material parameters are all constrained to be positive to improve simulation stability. Since the material parameters have physical meanings such as Young's modulus and spring stiffness, the non-negativity constraint specifies that the coarse material is made of traditional base materials.

Fitting in the Presence of Anisotropy If performed naively this optimization is non-linear because we must simultaneously solve for $\boldsymbol{v}_k$, the preferred direction of anisotropy. This can severely slow the fitting procedure, especially in cases where it would otherwise be a linear least squares problem (i.e if all fine-scale materials are Neo-Hookean or a similarly simple material model). To avoid this problem we first estimate all anisotropy directions, and then solve Equation 2.8 for the remaining material parameters. Our intuition is that anisotropy manifests itself as preferential stretching along a particular axis. To find this axis, we apply stretching forces to a block in a discrete set of directions uniformly sampled over a sphere. If the stretching force is close to the direction of anisotropy, then the amount of stretching deformation is reduced. For any given stretching direction $\mathbf{v}$, we apply a stretching force and compute the deformation gradient $\mathbf{F}$ of each quadrature point. Under $\mathbf{F}$, a unit length vector in direction $\mathbf{v}$ is stretched to a new length $l = \|\mathbf{F}\mathbf{v}\|$. The set of all 3D vectors $l\mathbf{v}$ forms an ellipse-like shape. We find the principal axes of the ellipse (via SVD) and use them as directions of anisotropy.

Regularization Since vastly different material assignments, ${}^0\mathcal{M}_k$, can produce the same coarse material, our naïve cost function (Equation 2.8) can produce very large parameter values and even non-physical negative ones. For example, consider a homogeneous material assignment at the high-resolution level. The same coarse material can be achieved by interleaving hard and soft materials at each fine element or by assigning a single, well chosen material to all fine elements. To overcome this, we add a regularization term to control the parameter ranges and prevent overfitting of the training samples. Our modified error function takes the following form:

$$\sum_{s=1}^{n_s} \left(V_s^0 - V^1\left(\boldsymbol{r}\left(\boldsymbol{u}_s^0\right), \boldsymbol{p}^1\right)\right)^2 + \lambda \sum_k (\boldsymbol{p}_k^1 - \boldsymbol{p}_k^0)^2, \tag{2.9}$$

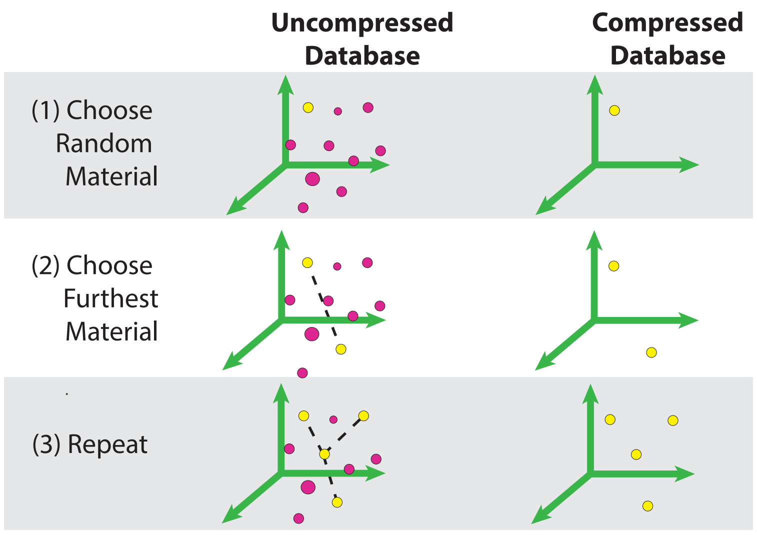


Figure 2-7: Database compression step adds coarse materials to a compressed database by a farthest point sampling strategy.

which prevents material parameters from deviating too far from $\mathcal{M}_k^0$. We chose the regularization constant $\lambda = 0.02$ for the results in this paper. In our experiments, since the base energy functions are linear with respect to the material parameters, the fitting problem can be solved by linear regression with regularization.

2.3.4 Database Compression

Given n materials in the palette, the number of material combinations in a $2\times2\times2$ block is n^8. In modern hardware, it is impossible to compute and store all material combinations even for a moderately-sized palette with 100 materials. In order to compress the number of materials stored in our coarse material database, we select a small number of representative material combinations and remove all others. We compare materials in a feature space. In order to construct coarse material feature vectors, we first select a common subset of deformations from all computed deformation samples. We then evaluate the potential energies of each coarse material at each deformation sample. The stacked vector of energies becomes a coarse material feature vector.

Since our base materials differ in stiffness by orders of magnitude, we take the logarithm to measure the difference in ratio. Let D be the L^2 norm of log-energies between the two materials given by

$$D(A, B) = \sqrt{\sum_s (\log(V_s^A) - \log(V_s^B))^2}, \tag{2.10}$$

where A and B denote two distinct coarse materials in the database. Given the distance metric, we can select k representatives materials using farthest point sampling Eldar et al. [1997]. We randomly choose an initial coarse material and then repeatedly select the material combination furthest away from any previous representatives – continuing until we obtain k representatives (Figure 2-7). This compression algorithm chooses k samples that equally

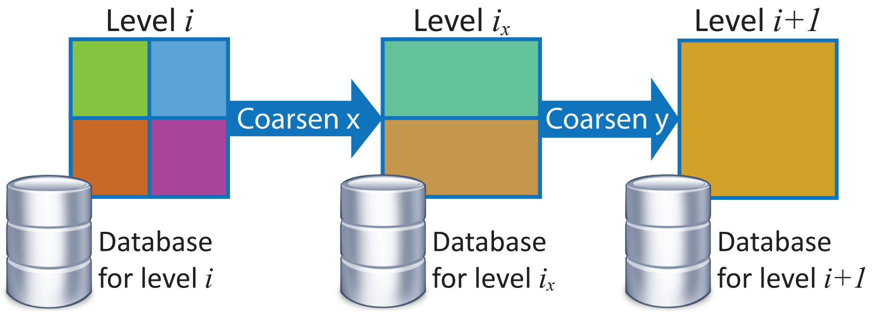


Figure 2-8: Hierarchical coarsening operates on one dimension at a time, performing clustering at each intermediate stage (here denoted i_x). This allows our compression algorithm to be applied aggressively, greatly reducing the number of energy samples we need for fitting material parameters.

cover the coarse material energy space, helping to preserve good behavior in our coarse simulations.

2.3.5 Hierarchical Coarsening

While one level of coarsening can yield significant speed-ups, DDFEM can also be applied hierarchically. As discussed in subsection 2.3.4, the exponential growth of coarse material palettes at each level makes it prohibitively expensive to perform fitting. We address this by changing our coarsening strategy. Instead of choosing $\mathcal{E}^0$ to be a $2{\times}2{\times}2$ block we choose it to be a $2{\times}1{\times}1$ block, which we coarsen. We construct an intermediate database of materials and compress. We then choose $\mathcal{E}^0$ to be a $1{\times}2{\times}1$ block, coarsen and compress, and finally a $1{\times}1{\times}2$ block, coarsen and compress. Intermediate compression greatly reduces the number of samples we need to generate in order to populate the material parameter database for the next coarsening level. It is important to note that our intermediate databases only store lookup tables which allow us to extract appropriate material IDs for the next coarsening stage. Material parameters need only be stored in the final database since it is these elements that are simulated.

2.4 Runtime Simulation

Once our coarse material database, $\mathcal{P}^1$, has been constructed we can use it to perform fast online coarsening. Initially, the user loads geometry which is embedded in a hexahedral grid for simulation. Prior to simulation we iterate over all $2{\times}2{\times}2$ blocks of hexahedral elements and perform mesh coarsening by replacing these 8 elements with a single coarse element. We perform a database lookup into $\mathcal{P}^1$, using the material ID numbers of the 8 original elements, to quickly retrieve the optimal coarse material for this coarse element. Database lookup is fast (even using our unoptimized, single-threaded implementation), and this is what makes DDFEM so appealing. We achieve significant simulation speed-up from coarsening, retain accuracy in the simulation, and reduce the cost of material coarsening at runtime to a negligible amount. Our material model can be used in any simulation algorithm suitable for non-linear elasticity. In our experiments, we use Coin-IpOpt Wächter and Biegler [2006] to implement static and dynamics simulations with tolerance (“tol” option) set to 0.5. We use Pardiso as our linear solver. For timing purposes, we limit Pardiso to single thread mode. The pseudo-code for static simulation is shown in Algorithm 1.

Algorithm 1 Static Simulation

```
1: repeat
2:     $\mathbf{f}$: global force vector
3:     $L$: triplet list for global stiffness matrix
4:     for each element e do
5:         compute elastic force $\mathbf{f}_e$
6:         add $\mathbf{f}_e$ to $\mathbf{f}$
7:     end for
8:     add external force $\mathbf{f}_{ext}$ to $\mathbf{f}$
9:     for each element e do
10:        $\mathbf{K}_e$: element stiffness matrix
11:        for each quadrature point q do
12:            compute stiffness matrix $\mathbf{K}_q$ at quadrature point
13:            $\mathbf{K}_e + = \mathbf{K}_q$
14:        end for
15:        append entries of $\mathbf{K}_e$ to $L$
16:    end for
17:    sort $L$ to get sparse stiffness matrix $\mathbf{K}$
18:    set entries in $\mathbf{K}$ and $\mathbf{f}$ for fixed vertices
19:    $\Delta\mathbf{x} = \mathbf{K}^{-1}\mathbf{f}$
20:    compute step size $h$ using line-search
21:    $\mathbf{x} + = h\Delta\mathbf{x}$
22: until convergence
```

2.5 Results and Discussion

All the results shown here are simulated using nonlinear constitutive models at the fine scale. This and coarsening speed are the key differentiating factors between DDFEM and other coarsening algorithms such as Nesme et al. [2009] and Kharevych et al. [2009]. Our database starts with three Neo-hookean base materials with Young’s modulus $1e5, 1e6, 1e7$ and Poisson’s ration 0.45. For comparison with 3D-printed objects, we used two base materials with measured Young’s moduli. We use 500 force directions, and sample 5 magnitudes in each direction, resulting in 2500 force samples for each material combination. In addition, we generate 500 stretching samples for computing the direction of anisotropy. During fitting, we use shear modulus and Lamé’s first parameter, as well as the spring stiffness. We repeat the same process for the second level of coarsening, using 6561 materials in the first level as base materials. We select 400 representatives at each intermediate level.

Example	grid size	rel sp	time/iter	iters	error
Pushing(0)	16×16×16	1.0	1.010	5	-
Pushing(1)	8×8×8	11.5	0.087	5	8.91e-4
Pushing(2)	4×4×4	31.4	0.032	5	1.36e-2
Bending(0)	8×32×8	1.0	0.270	28	-
Bending(1)	4×16×4	12.6	0.028	22	5.60e-2
Bending(2)	2×8×2	22.7	0.015	22	8.88e-2
Twisting(0)	8×32×8	1.0	0.300	16	-
Twisting(1)	4×16×4	15.2	0.031	10	1.56e-2
Twisting(2)	2×8×2	20.7	0.019	12	3.28e-2
Buckling(0)	128×8×16	1.0	8.85	32	-
Buckling(1)	64×4×8	50.1	0.28	20	7.24e-3
Buckling(2)	32×2×4	331.8	0.12	7	3.14e-2
Fibers(0)	32×100×32	1.0	193.85	17	-
Fibers(1)	16×50×16	51.2	4.95	13	2.94e-2
Fibers(2)	8×25×8	489.5	0.96	7	4.26e-2
Bridge(0)	56177	1.0	43.44	14	-
Bridge(1)	9727	8.4	4.88	15	4.39e-3
Bridge-arch(0)	65684	1.0	54.99	3	-
Bridge-arch(1)	11695	8.1	7.84	3	3.68e-4
George(0)	46152	1.0	52.19	23	-
George(1)	6755	16.4	3.49	21	2.86e-2
George-bone(0)	46152	1.0	41.35	12	-
George-bone(1)	6755	13.2	2.70	14	2.99e-2

Table 2.2: Relative performance, absolute performance in seconds and average vertex error relative to the bounding box size for full-resolution and coarsened simulations. Relative performance illustrates the performance increase gained by coarsening with respect to the time taken for the high-resolution static simulation to converge. Bracketed numbers after each example name indicate the number of coarsening levels with 0 indicating the high-resolution simulation. All computation times are recorded using Coin-IpOpt running in single threaded mode on a 2.5 GHz Intel Core i7 processor.

2.5.1 Database

One advantage of our compact coarse material representation is the small amount of storage it requires. In fact we require only $6 \times 8 = 48$ floating-point values for each material at the first coarsening level and $6 \times 64 = 384$ values for the second level. (For each finer element, ${}^0\boldsymbol{p}$ contains 2 material moduli plus $C, \boldsymbol{v}$.) For further recursive levels, we can limit ourselves to 320 values per material. Our current 3 material database is 4 megabytes in size.

2.5.2 Simulation Results

We show results from elastostatic simulations performed using DDFEM. We also demonstrate its performance advantages over high-resolution simulations. We render wire frames to show the discretizations of the high-resolution and coarse meshes. We first show examples of two simple simulations, the pushing and twisting of a rectangular object with heterogeneous, layered material distribution (Figure 2-9). Note that in all cases DDFEM qualitatively matches the behavior of the high-resolution simulation. We also compare the performance

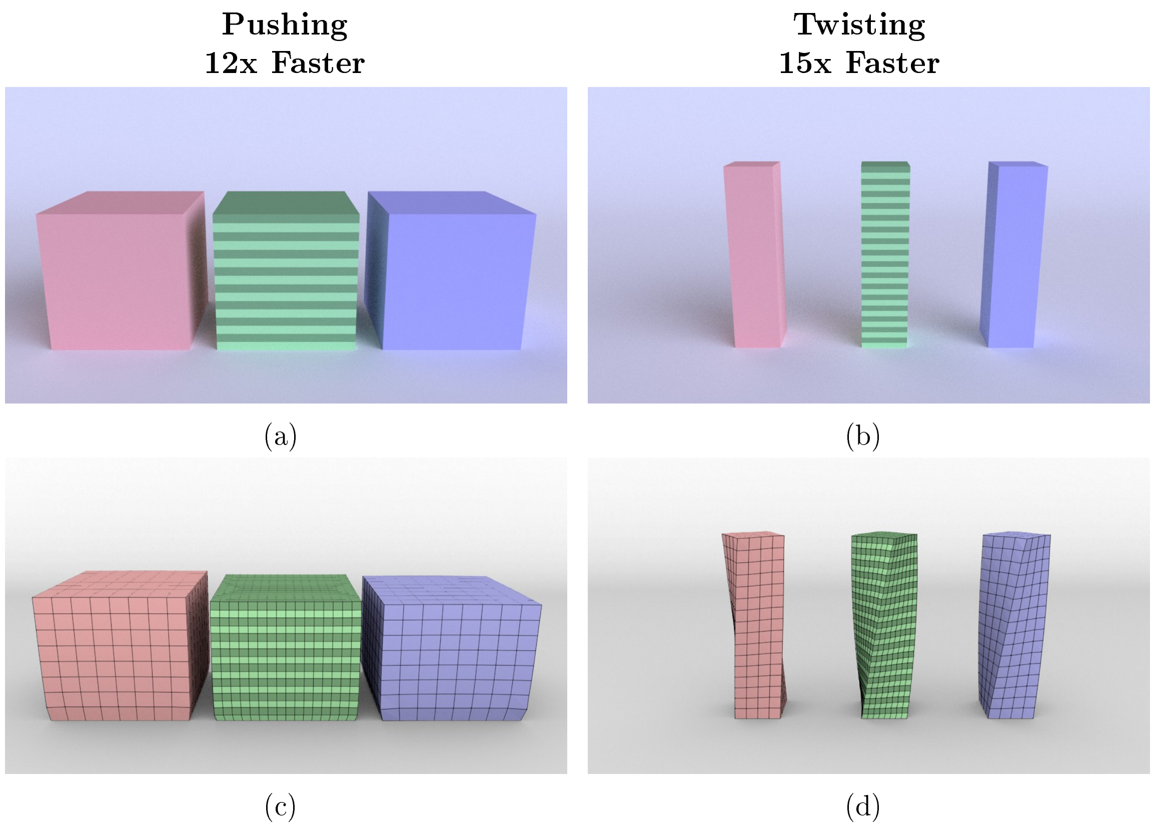


Figure 2-9: Examples of pushing a cube (a - Initial State, c - Compressed) and twisting a bar (b - Initial State, d - Compressed), both with heterogeneous material distribution. We compare DDFEM to Naïve Coarsening and the ground-truth, High-Res Simulation. We render wire frames to show the simulation meshes.

of DDFEM to a naïve coarsening method that uses the material properties from 2×2×2 element blocks of the high-resolution simulation mesh at each corresponding quadrature point. In our supplemental video we compare to a second baseline model which averages material parameters inside each coarse element. This average model is less accurate than the Naïve model in all cases.

Naïve approaches often exhibit pathological stiffness for heterogeneous materials (illustrated by the lack of compression of the box and lack of twisting of the bar) Nesme et al. [2009]. In these cases, DDFEM yields good speed ups while maintaining accuracy. For a single level of coarsening we achieve *8 times or greater* speed ups for all examples. Performance numbers and mean errors are listed in Table 2.2. Since the fine simulation and the coarse simulation have different numbers of vertices, we create a fine mesh from the coarse simulation by trilinearly interpolating the fine vertices using the coarse displacements. The errors are measured by computing the average vertex distance relative to the longest dimension of the bounding box in rest shape. We also examine the behavior of DDFEM during bending (Figure 2-10). Yet again the naïve coarsening method completely fails to capture the behavior of the high-resolution result, while DDFEM offers a much better approximation.

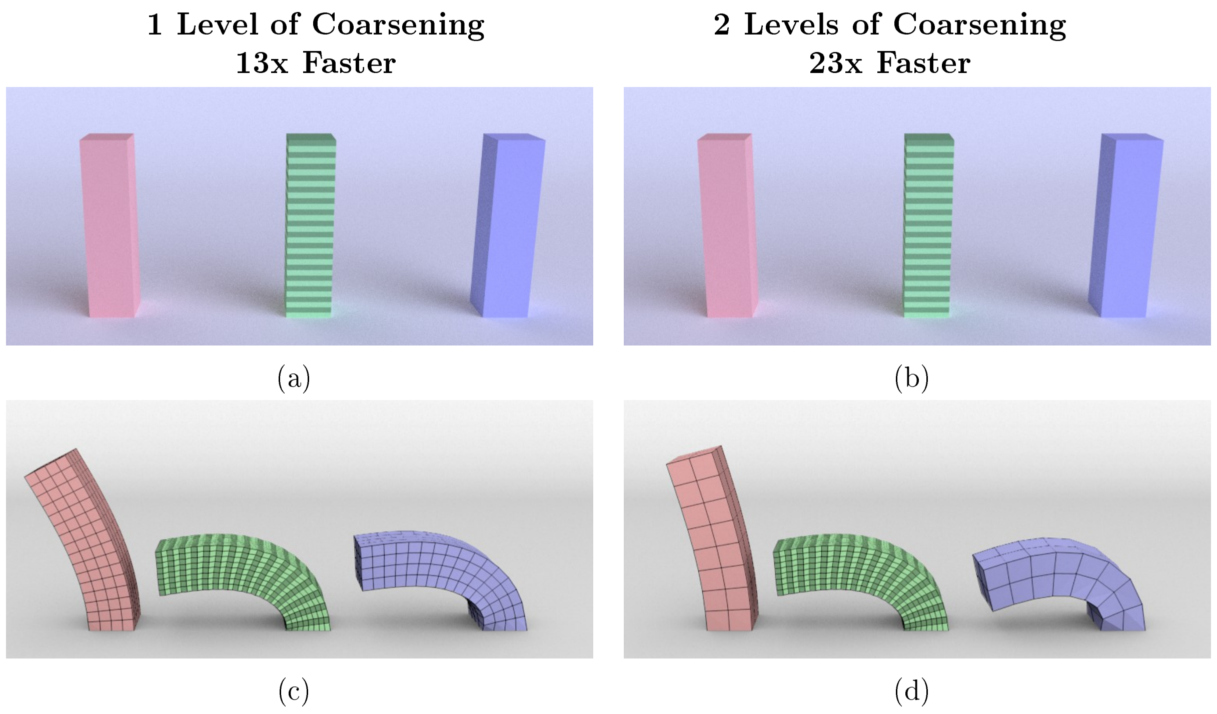


Figure 2-10: Bending a heterogeneous bar: We compare DDFEM to Naïve Coarsening and a High-Res Simulation. Subfigures (a, c) show comparison for 1 level of coarsening, and (b, d) show 2 levels of coarsening. The naïve coarsening approach results in a much stiffer behavior, whereas our fitted model more closely approximates the fine model.

Performance Analysis Table 2.3 shows time spent in quadrature evaluation versus in solver at coarse and fine levels. We use 8 quadrature points for the first level of coarsening and 64 for the second level. The time for computing the local stiffness matrix for one element increases from 0.1ms to 1ms. In the second level, the speedup comes from the reduced number of elements over which to perform quadrature and the time required for the linear solver.

To further investigate the performance of our coarsened simulations we replaced Pardiso with an assembly-free Jacobi-preconditioned conjugate gradient (CG) linear solver and used this to simulate our George-bone test case. While the overall runtime of the high-resolution simulation increased from 496 to 2082 seconds (most likely do to the unoptimized nature of our solver) our coarsened model achieved 20x and 67x speedups using one level and two levels of coarsening respectively. One might expect no benefit from the second level of coarsening since the number of quadrature points remains constant. However, the number of CG iterations is roughly proportional to the number of vertices in the simulation mesh and thus the coarse model converges more quickly (Figure 2-13). Since our coarse material models are not restricted to use a fixed number of quadrature points, one could design coarse models that are more tailored towards assembly-free solvers by reducing the number of quadrature points and simplifying the strain energy expressions.

Complex material behavior DDFEM can capture the gross behavior of complex, spatially-varying material distributions. Figure 2-11 shows the results of applying DDFEM to a non-

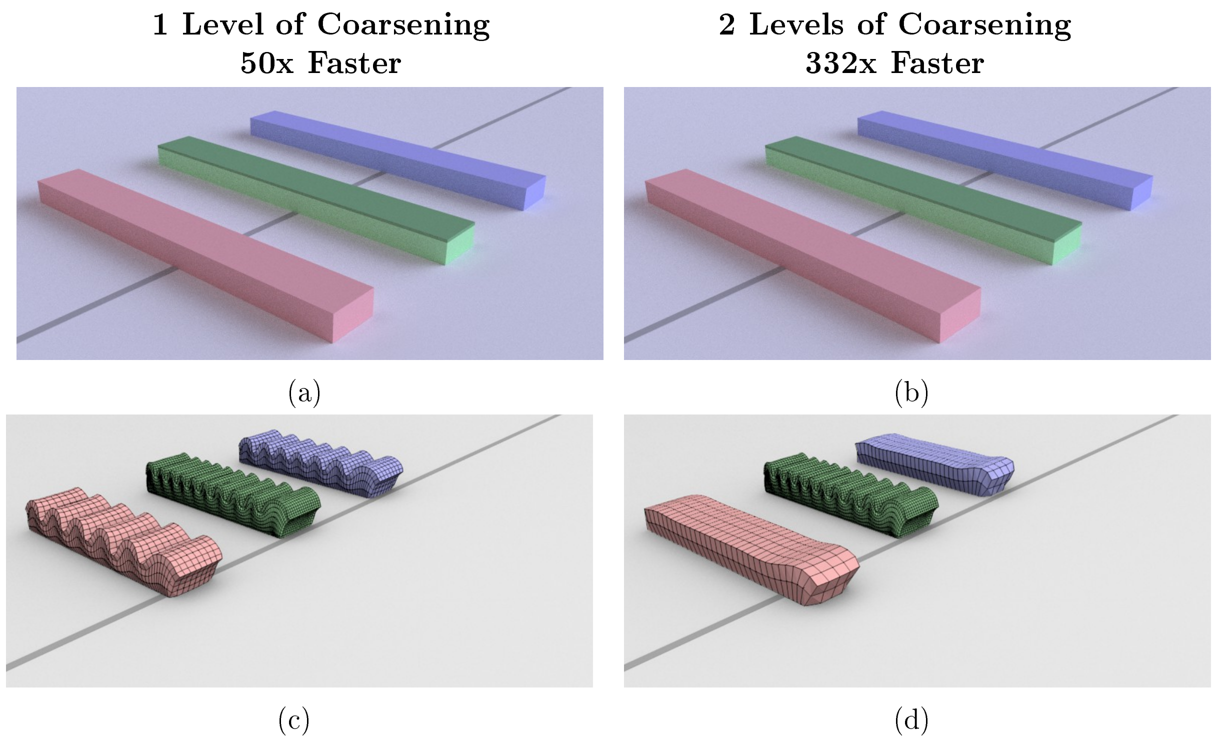
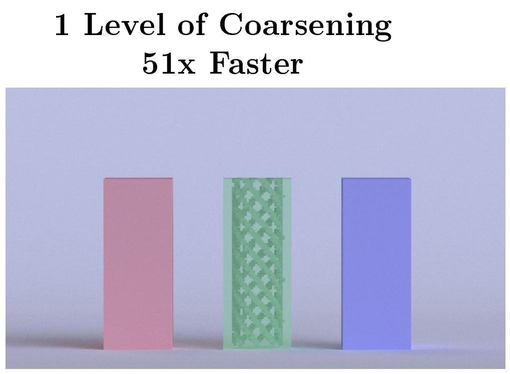


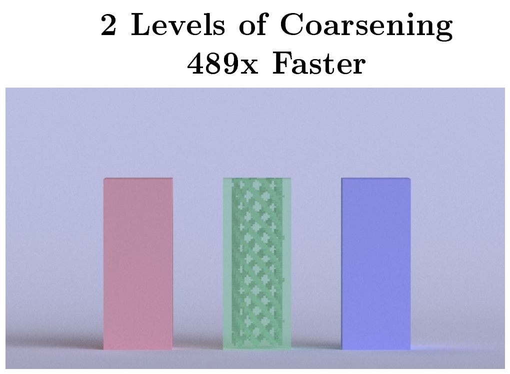
Figure 2-11: Compressing a heterogeneous slab using Naïve Coarsening (1 level and 2 levels of coarsening), DDFEM (1 level and 2 levels of coarsening) and a High-Res Simulation. The top, darker layer is stiffer, causing the object to buckle. The bottom vertices are constrained to stay on the floor. Figure (a,b) shows the slabs before compression, figure (c,d) shows the slabs after compression and figure. Notice that, after 1 level of coarsening, Naïve Coarsening neither compresses nor buckles as much as either DDFEM or High-Res Simulation. After 2 levels of coarsening, the buckling behavior is lost. The Naïve Coarsening fails to capture the compressive behavior of High-Res Simulation, whereas DDFEM does.

linearly elastic slab with a stiff "skin." The bottom of the slab is constrained to slide along the ground with one end fixed. When force is applied to the free end of the slab, buckling occurs. Somewhat obviously, DDFEM cannot replicate the frequency of the high-resolution buckling pattern due to the coarseness of the simulation mesh. However, it correctly captures the gross behavior of the bar and approximates the overall amount of compression well. Figure 2-11 also shows a comparison with 2nd level coarsening. In this case, the overall compression of the bar is still captured accurately. For this example DDFEM affords *50 times* (1 level of coarsening) and *332 times* (2 levels of coarsening) performance improvements over the high-resolution simulation. The artificial stiffness of the naïve model can be seen in the reduced buckling and compression when compared to DDFEM at both coarsening levels.

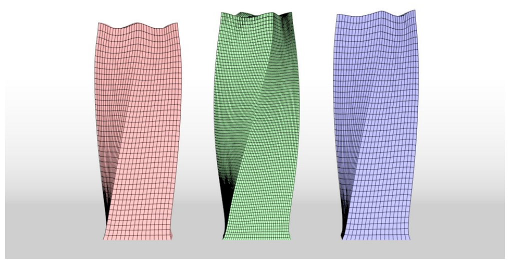
Anisotropic material distribution Next we explore the ability of DDFEM to handle highly anisotropic material distributions (Figure 2-12). Specifically, we embed a helical set of stiff fibers in a soft, non-linearly elastic matrix. Pulling on the object induces a twisting. Again, at one coarsening level DDFEM captures this anisotropic behavior well, much better than the naive approach, and gains a *51 times* speed up over the high-resolution simulation.



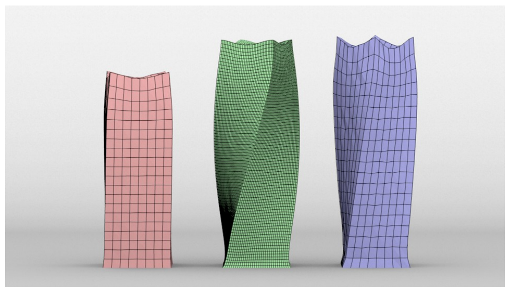
(a)



(b)



(c)



(d)

Figure 2-12: Simulating a bar with an embedded set of fibers using Naïve Coarsening, DDFEM and a High-Res Simulation. Note that DDFEM captures the characteristic twisting motion of the bar better than Naïve Coarsening. (a,b) shows the initial state of both bars while (c,d) shows the deformed state after pulling on the top of the bars.

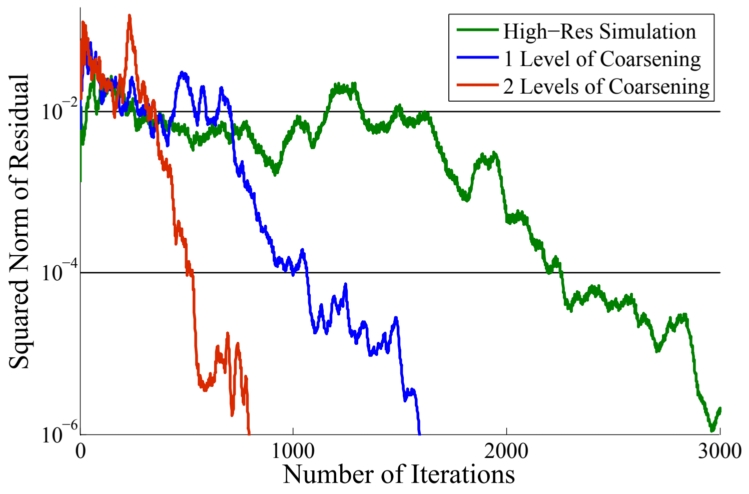


Figure 2-13: Comparison of CG iterations on high-resolution and coarsened meshes of the George-bone example. The squared residual is measured as $||\mathbf{Kx} - \mathbf{f}||^2$. We observe that CG converges much faster on coarse meshes.

Example	**grid size**	**quad/iter** (12)	**time/iter** (2-21)	**iters**
Bending(0)	8×32×8	0.11	0.27	28
Bending(1)	4×16×4	0.015	0.028	22
Bending(2)	2×8×2	0.014	0.015	22
Buckling(0)	128×8×16	1.0	8.8	32
Buckling(1)	64×4×8	0.11	0.28	20
Buckling(2)	32×2×4	0.10	0.12	7
Fibers(0)	32×100×32	10.0	193	17
Fibers(1)	16×50×16	1.0	4.9	13
Fibers(2)	8×25×8	0.68	0.96	7

Table 2.3: Portion of time in seconds used by quadrature computation during static simulation in seconds. Bracketed numbers indicate corresponding lines in Algorithm 1.

Worth noting is that the DDFEM bar is slightly softer in the y-direction. This kind of inaccuracy should be expected. Since our method builds a low-dimensional approximation of a potential energy function we cannot hope to accurately reproduce the complete behavior of the high-resolution simulation. What is important is that DDFEM captures the salient global behavior, in this case, the twisting of the bar.

Geometry and material design We present three examples of using DDFEM for geometry and material design. In the first example, we edit the material composition of the sole of a running shoe in order to stabilize it. Figure 2-14 shows the effect of the three material edits as well as relative speed up achieved over the full resolution simulation and coarsening time. DDFEM performance is always an order of magnitude more than that of the high-resolution simulation, and, most importantly, our coarsening times are on the order of milliseconds. We stress that our current implementation is completely single threaded and that coarsening, which in our case involves a simple database lookup, is inherently parallel. In the second example, we add a supporting arch to a bridge. Prior to the addition of the support structure, the bridge sags catastrophically. The fast coarsening of DDFEM allows us to achieve an 8 fold increase in simulation performance using a single coarsening pass. In the third example, we add a rigid skeleton to a deformable character (George) in order to control his pose. Here our single threaded, data-driven coarsening only takes 200ms.

Dynamics Though the examples shown in this paper focus on static analysis, DDFEM is equally applicable to dynamic simulations. At its core, DDFEM simply supplies new, more accurate material models for use during simulation. This makes the method useful for accelerating various animation tasks as well. In the accompanying videos we show a dynamic simulation of our fiber embedded bar, computed using a standard linearly-implicit time integrator.

2.5.3 Fabricated Results

Finally, we test the accuracy of our simulation against fabricated results, created using a Stratysys Object Connex 500 multimaterial 3D printer. We fabricated a bar with embedded

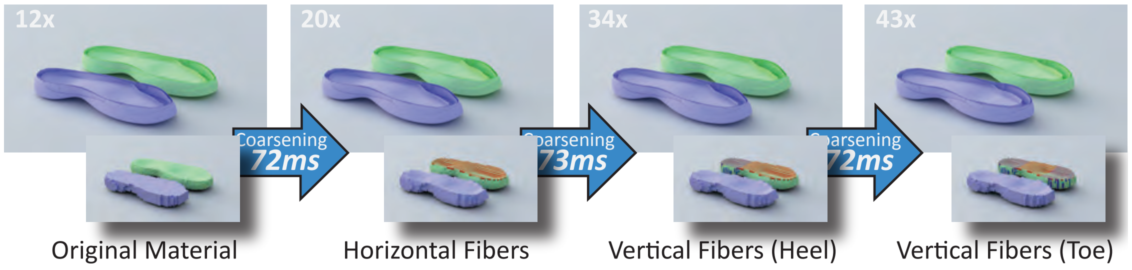


Figure 2-14: Designing a shoe sole: We compare the performance of DDFEM to that of High-Res Simulation in the context of a material design problems. Large images show the effect of material changes on the sole of the shoe, which is being deformed under a "foot-like" pressure field. Inset images show the materials assigned to the shoe sole and the embedded finite element simulation mesh. Numbers within arrows show coarsening times between editing steps and the numbers in the upper left corner of each image show the relative performance of DDFEM to High-Res Simulation. While the DDFEM sole is made up of many coarse material, we display it as a single color to distinguish it from the High-Res Simulation.

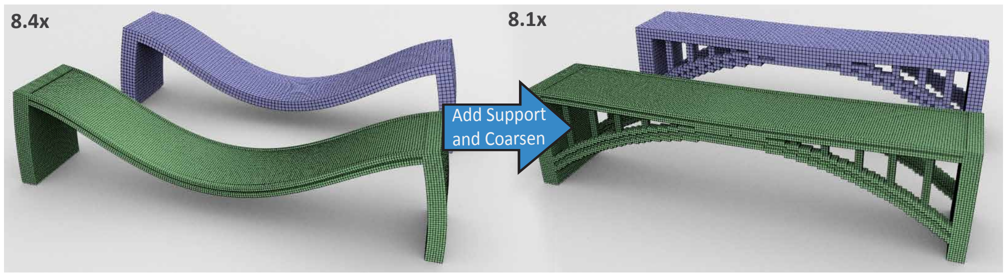


Figure 2-15: Accelerating geometry change: We repair a structurally unsound bridge by adding a supporting arch (8x faster).

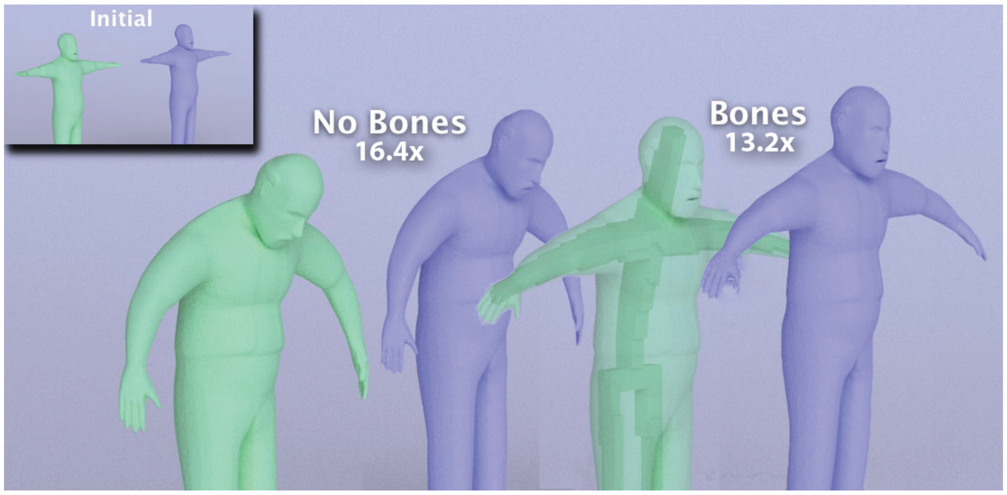


Figure 2-16: Correcting George's posture using a rigid skeleton (High-Res Simulation and DDFEM).

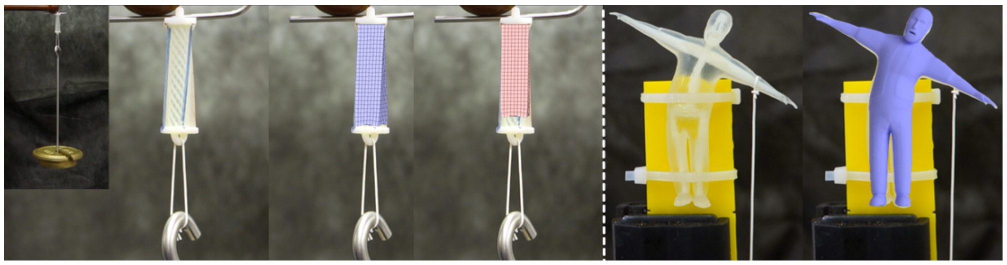


Figure 2-17: A comparison of DDFEM (2 levels of coarsening) to real world deformations of 3D printed, multi-material designs. DDFEM captures the twisting behavior of an anisotropic bar much more accurately than Naïve Coarsening. Similarly DDFEM accurately predicts the deformation of our heterogeneous George character.

helical fibers as well as our George character and applied specified loads to both. We show qualitative comparison of the deformed configurations of these real-world examples to our simulated results (*2 levels of coarsening*-Figure 2-17). Note that the simulation does an excellent job of predicting the deformed configuration of both objects.

2.6 Limitations and Future Work

Because DDFEM relies on a database compression step to combat the combinatorial explosion of coarse materials, accuracy is heavily influenced by the set of representative coarse materials. Finding a better way to select coarse structures is an interesting area of future work. Second, in our attempt to make our method geometry independent, some accuracy when dealing with partially filled boundary finite elements is sacrificed. Adding a parameterized boundary representation to the method, in order to more correctly handle non-axis aligned boundary conditions, could also be explored. Third, the method acts on discrete materials. While we believe that this is reasonable, considering the way that engineers and designers approach material design, a method that coarsens continuous spaces of non-linear materials could be beneficial.

Many avenues of future work are promising. First, one could explore topologically aware meshing (i.e. in the same vein as Nesme et al. [2009]) to allow better handling of models with large empty regions. In fact shape function learning approaches, such as Nesme et al. [2009] could be combined with our material learning approach to produce even more accurate simulations. Including these shape functions in our database could, for instance, allow us to capture the wrinkles in our buckling example. Second, extending DDFEM to more complex material models, such as those involving plasticity, would be a useful exercise. Third, DDFEM can be combined with an adaptive voxel grid as well as other dynamic meshing approaches to obtain further speed-ups. Finally, exploring hierarchical solvers based on DDFEM coarsening is a very attractive direction. Solvers such as multigrid methods rely on good coarse approximations to accelerate fine scale simulations. Using DDFEM for these approximations could improve the convergence rate, and thus the performance of such algorithms.

Chapter 3

Computational Design with Microstructures

Many engineering problems are formulated as high level objectives such as the ability to support certain weight, optimal tradeoffs between compliance and mass, minimal deformation under temperature changes, etc. A popular approach to design such structures is topology optimization. Topology optimization generally refers to discretizing the object of interest into small elements and optimizing the material distribution over these elements such that the functional goals are satisfied Bendsøe and Sigmund [2004]. Traditionally, topology optimization focuses on designs made of homogeneous materials and is concerned with macroscopic changes in the object geometry. With the advent of multi-material 3D printing techniques, it is now possible to assign materials at a much higher resolution, allowing much finer designs and improved functional performances. Unfortunately, standard techniques for topology optimization do not scale well and they cannot be run on objects with billions of voxels. This is because the number of variables to optimize increases linearly with the number of cells in the object. Since many current 3D printers have a resolution of 600DPI or more, a one billion voxel design occupies only a 1.67 inch cube.

Most previous algorithms use a single material property variable such as density or material stiffness in each cell, for which analytical formulas describing the property bounds exist Allaire and Kohn [1993]. On the contrary, optimizing the structure and material distribution of an object in a high dimensional material property space remains an open problem. In this work, we propose a new computational framework for topology optimization with microstructures that supports design spaces of multiple dimensions. We start by computing the gamut of the material properties of the microstructures by alternating stochastic sampling and continuous optimization. This gives us a *discrete* representation of the set of achievable material properties, from which we can construct a *continuous* gamut representation using a level set. We then reformulate the topology optimization problem in the continuous space of material properties and propose an efficient optimization scheme that finds the optimized distributions of multiple material properties simultaneously inside the gamut. Finally, in order to obtain fabricable designs, we map the optimal material properties back to discrete microstructures from our database.

Our general formulation can be applied to a large variety of problems. We demonstrate its efficacy by designing and optimizing objects in different material spaces using isotropic, cubic and orthotropic materials. We apply our algorithm to a diverse range of functional objectives such as minimal compliance and target strain distribution. Furthermore, our approach utilizes the high-resolution of current 3D printers by supporting designs with one trillion voxels. We fabricate several of our designs to demonstrate practical applications of our method in 3D printing. The main contributions of our work can be summarized as follows:

- We present a fully automatic method for computing the space of material properties achievable by microstructures made of a given set of base materials.
- We propose a generic and efficient topology optimization algorithm capable of handling objects with a trillion voxels. The key of our approach is a reformulation of topology optimization to work directly on continuous variables representing the material properties of microstructures. This allows us to greatly reduce the problem scale for it to be efficiently solved with state of the art solvers.
- We validate our method on a set of test cases and demonstrate its versatility by applying it to various design problems of practical interest.

3.1 Overview

Given as input a set of base materials, an object layout, and functional objectives, the goal of our system is to compute the material distribution inside the object in order to optimize these functional objectives. In our approach, we do not solve the problem directly, instead we work with microstructures made of the base materials. The complete pipeline of our system has three stages (Figure 3-1).

Material Space Precomputation In the first stage, we estimate the gamut of material properties covered by all possible microstructures made by spatial arrangement of base materials. Since exhaustively computing the properties of all these microstructures is, in practice, intractable, we progressively increase the material space by alternating a stochastic search and a continuous optimization. The first step introduces discrete changes in the materials of the microstructures and allows emergence of new types of microstructures. The second step allows to locally push the material space boundaries by refining the microstructure shapes. After completing this stage, we obtain a discrete representation of the space of material properties and the mapping between these properties and the corresponding microstructures.

Gamut-based Continuous Topology Optimization In the second stage, we construct a smooth continuous gamut representation of the material property space by using a level set field. We define our topology optimization problem directly in this space. Our approach minimizes the objective function over possible material parameters while asking for strict satisfaction of the physics constraints – typically, the static equilibrium – as well as the strict satisfaction of the physical parameter bounds. Taking advantage of our gamut representation as a level set, we formulate this last constraint as limiting the material properties to stay on

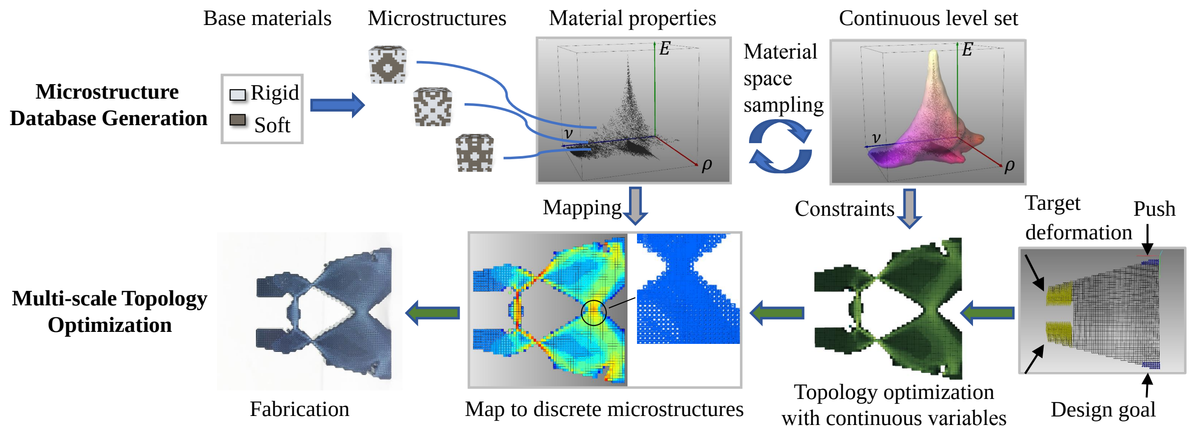


Figure 3-1: Algorithm overview. We start by precomputing the gamut of material properties that can be achieved with all material microstructures of a given size. Next, we run our topology optimization algorithm that optimizes the material properties of the object within this gamut such as to minimize some functional objective. Finally, we map the optimal continuous material properties back to microstructures from our database to generate a printable object.

the negative side of the level set. This guarantees that the material properties that we use in the optimization are always physically realizable.

Fabrication-oriented Microstructure Mapping In the last stage, we generate a printable result by replacing each cell in the object layout with a microstructure whose material properties are the closest to the continuous material assignment resulting from the optimization. We also take into account the boundary similarity across adjacent cell interfaces to improve the connectivity between microstructures. This results in a high-resolution, multi-material model with optimized functional specifications.

3.2 Material Space Exploration

The first step in our pipeline is to determine the range of achievable physical properties when combining the base materials into microstructures at a given resolution. Computing the mechanical properties of microstructures arranged in periodic tilings can be performed using physics-based simulation. We use the homogenization theory to compute the elastic properties of each microstructure Allaire [2012], Panetta et al. [2015], Schumacher et al. [2015]. However, while inferring the homogenized properties of individual microstructures is not particularly challenging, analyzing the space covered by *all* combinations of base materials is much more difficult due to the combinatorial explosion in the number of possible material arrangements. As an example, $16 \times 16 \times 16$ lattices made of only two materials corresponds to 2^{4096} microstructures. Exhaustively simulating all microstructures is clearly infeasible in practice. To address this issue, two possible avenues can be pursued:(i) sample the space of microstructures, (ii) take advantage of the continuity between material parameters of the individual voxels and macroscopic properties of the microstructures in order to generate new

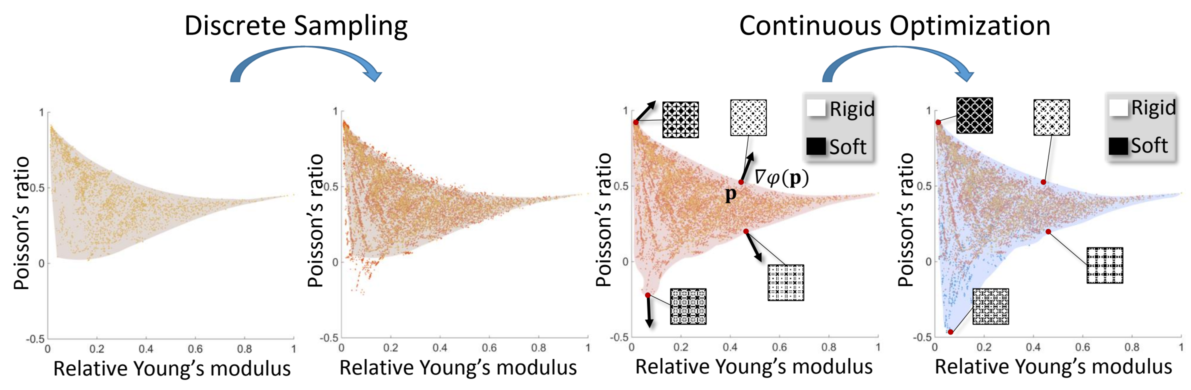


Figure 3-2: One cycle of computing the microstructure gamut. Given a set of samples, we compute a signed distance function approximating the material gamut (*left*) and randomly perturb microstructures lying near the boundary to provide new seeds to the continuous algorithm (*middle left*). We then update the distance field and use the gradient of the signed distance function at the the boundary to define new target material points (*middle right*). These target material points are used in a continuous optimization that generates new samples (*right*).

microstructures with desired properties. The second option is effective in reaching locally optimal values in the material property space. However, the continuous function that maps the material assignment to material properties is nonlinear. In particular, very different microstructures can correspond to the same point in the material property space. Additionally, since the ratio of materials in each cell is bounded between zero and one, the continuous optimization converges slowly or stops moving when material distributions in many cells are at the lower or upper bound. Being able to jump out of a local optimum and discovering different variants is important in order to provide new exploration regions. We combine the strength of these two approaches in a scheme that alternates between a stochastic search and a continuous optimization. We provide the technical details in the rest of this section.

3.2.1 Discrete Sampling of Microstructures

We aim at sampling the space of material assignments, i.e. microstructures, in a way that maximizes the number of samples whose material properties lie in the vicinity of the material gamut boundaries. We do not draw all samples at once but progressively enrich the database of microstructures as we refine our estimation of the material gamut boundaries. This sampling strategy is motivated by the observation that a small change in the material assignment of a microstructure generally – but not always – translates to a small change of its material properties. By modifying microstructures located near the current boundaries of the material property gamut, we are likely to generate more structures in this area, some of which will lie outside of the current gamut.

Given a population of microstructures to evolve, we generate new samples from each microstructure by changing adding or subtracting random beams. To rationalize computational resources, we want to avoid revisiting the same voxel twice. But we do not want to privilege any particular order either. Ritchie et al. [2015] recently presented a Stochastically-Ordered

Sequential Monte Carlo (SOSMC) method that provides a suitable approach. In SOSMC, a population of particles (microstructures) corresponding to instances of a procedural program (the assignment of materials to the voxels of the microstructures) are evolved so as to represent a desired distribution. During this process, the programs are executed in a random order and particles are regularly scored and reallocated in regions of high probability. In our particular settings, we use the scoring function

$$s(\mathbf{p_i}) = \frac{\Phi(\mathbf{p_i})}{D(\mathbf{p_i})} \times \frac{1}{D(\mathbf{p_i})}, \tag{3.1}$$

where $\Phi(\mathbf{p_i})$ is the signed distance of the material properties of particle i to the gamut boundary (see Section 3.2.3) and $D(\mathbf{p_i})$ is the local sampling density at the location $\mathbf{p_i}$. We define the sample density as

$$D(\mathbf{p_i}) = \sum_{\mathbf{k}} \phi_{\mathbf{k}}(\mathbf{p_i}) \ , \tag{3.2}$$

where $\phi_k(\mathbf{p}) = \left(\mathbf{1} - \frac{||\mathbf{p}-\mathbf{p_k}||_2^2}{\mathbf{h^2}}\right)^{\mathbf{4}}$ are locally-supported kernel functions that vanish beyond their support radius h, set to a tenth of the size of the lattice used for the continuous representation of the material gamut (see Section 3.2.3).

The first term in Equation 3.1 favors microstructures located near the gamut boundary. The normalization by D allows us to be less sensitive to the local microstructures density and to hit any location corresponding to the same level-set value with a more uniform probability. The second product is used to additionally privilege under-sampled areas.

Particles are resampled using systematic resampling scheme [Douc, 2005] that is also used to initiate the population of particles. These particles are then evolved by adding and subtracting beams with random sizes. The sampled structures are then cleaned by removing unsupported components and filling enclosed cavities.

3.2.2 Continuous Optimization of Microstructures

The goal of the continuous optimization is to refine the geometry of the microstructures located at the boundary of the gamut in order to further expand the gamut along the normal directions (Figure 3-3). We start continuous sampling by selecting a subset of microstructures lying on the boundary of the gamut. The discrete structures are mapped to continuous values close to 0.5. We used 0.5 ± 0.3 in our experiments. Doing so allows the topology optimization algorithms to move freely in the first steps and discover new structures. We show an example of reducing the Poisson's ratio of an initial structure in Figure 3-3. In the plot, the initial Poisson's ratio is close to 0.4 since the starting point is similar to a homogeneous block.

For each starting structure, we identify target material parameters using the gradient of the level set Φ at the initial discrete sample point $\mathbf{p}$ (see Section 3.2.3) defined by $\mathbf{q} = \mathbf{p} + \nabla\Phi(\mathbf{p})$. We translate this target material parameters into an elasticity tensor $\mathbf{C}_0$ and density ρ_0. Here ρ is the ratio of the two base materials in the microstructure.

Note that our problem formulation does not restrict us to a particular topology optimization algorithm or material distribution parametrization. We have experimented with two objective functions that worked equally well for our purposes. The first objective uses an

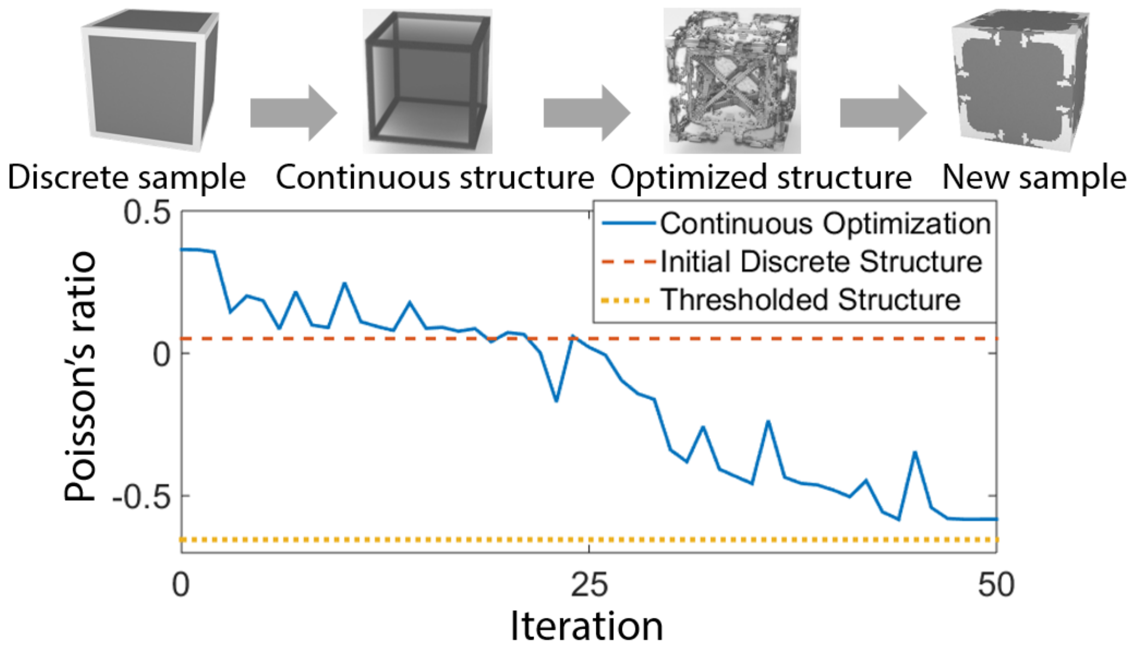


Figure 3-3: The continuous sampling step uses topology optimization to expand the gamuts by refining existing structures. A good starting point is necessary for the optimization to find better solutions. We use discrete microstructures near the boundary to initialize topology optimization. At convergence, we threshold the values to obtain a new discrete sample.

energy based formulation Xia and Breitkopf [2015a] to compute and optimize the elasticity tensor directly. At a high level, the optimization problem is

$$\arg\min_{\mathbf{x}} f(\mathbf{x}) = (\mathbf{C}(\mathbf{x}) - \mathbf{C}_0)^2 + w_\rho(\rho - \rho_0)^2, \quad \rho = \sum_i x_i, \tag{3.3}$$

where $\mathbf{x}$ is the ratio of materials in each cell, and w_ρ controls the weighting between the displacement term and the density term. Xia and BreitKopf developed parameter heuristics to optimize for difficult cases such as negative Poisson's ratio structures. We naturally arrive at structures with negative Poisson's ratio without the parameter varying step in Xia and Breitkopf [2015a] since our sampling strategy allows us to explore a wide variety of initial designs.

The second objective is formulated using harmonic displacements [Kharevych et al., 2009, Schumacher et al., 2015] $\mathbf{G}$ instead of the elasticity tensor directly. $\mathbf{G}$ is a 6×6 symmetric matrix where each row corresponds to a strain in vector form. We use the target elasticity tensor $\mathbf{C}_0$ to compute the target harmonic displacements matrix $\mathbf{G}_0$ and minimize the objective function:

$$f(\mathbf{x}) = (\mathbf{G}(\mathbf{x}) - \mathbf{G}_0)^2 + w_\rho(\rho - \rho_0)^2. \tag{3.4}$$

This objective matches soft structures more accurately since entries of G are inversely proportional to material stiffness.

Following the work by Andreassen et al. [2014], we use the method of moving asymptotes (MMA) Svanberg [1987]) to optimize the objectives using an implementation provided in the

NLOPT package [Johnson, 2014]. We run at most 50 iterations since it usually converges to a solution within 20-30 steps (Figure 3-3). MMA makes large jumps during the optimization while keeping track of the current best solution, thus causing the oscillation of the objective value. To force continuous material ratios towards discrete values, we experimented with the SIMP model with the exponent set to 3 and the Hashin-Shtrikman bound for isotropic materials described by Bendsøe and Sigmund [1999].

Either interpolation allows us to threshold the final continuous distribution and obtain a similar discrete sample. We tolerate small deviations introduced by the thresholding since our goal is to obtain a microstructure lying outside of the gamut rather than reaching a particular target. In practice, we observed that the material properties of the final discrete structures often did not change significantly after the thresholding step.

3.2.3 Continuous Representation of the Material Gamut

We represent the gamut of material properties using a signed distance field that is computed from the material points associated to the sampled microstructures. First, we normalize each coordinate p_i of $\mathbf{p}$ to constrain the scope of the level set to an n-dimensional unit cube. Then we compute the level set values on the cell centers of an n-dimensional Cartesian grid that encloses this unit cube. We draw inspiration from the methods for surface reconstruction used in particle fluid rendering Ando et al. [2013], Bhatacharya et al. [2011], Zhu and Bridson [2005] and extend it to n dimensions. In this case, a signed distance field is generated from a set of points by evaluating an implicit distance function Φ at each point $\mathbf{p} \in \mathcal{M}$. We initialize the signed distance field using the implicit function $\Phi(\mathbf{p}) = ||\mathbf{p} - \bar{\mathbf{p}}|| - r$ from Zhu and Bridson [2005] where $||\cdot||$ is the Euclidean distance between two points in $\mathcal{M}$, and $\bar{\mathbf{p}}$ is the average position of the neighboring points of $\mathbf{p}$ within a range of $2r$. Note that the signed distance is initially defined only near the boundary of the gamut. In order to sample the distance on the entire domain, we propagate the 0-level set surface using the fast marching algorithm and solve an explicit mean curvature flow problem defined as $\partial\Phi/\partial t = \Delta\Phi$ [Osher and Fedkiw, 2006]. Having a continuous representation of the gamut of materials achievable by the microstructures, we can now reformulate the topology optimization problem directly in the material space.

3.3 Discovery of Extremal Microstructure Families

Microstructures can exhibit remarkable physical properties that extend beyond the properties of their constituent materials. Many microstructure types have been developed to demonstrate applications in mechanics [Kadic et al., 2012, Li and Gao, 2016, Meza et al., 2014, Milton and Cherkaev, 1995, Wang et al., 2016, Zheng et al., 2014], acoustics [Fang et al., 2006, Li et al., 2009], and electromagnetics [Schurig et al., 2006, Shalaev, 2007]. These microstructures are typically designed by domain experts using time and labor intensive manual processes. These designs are often programmable in the sense that they have a small number of parameters to generate a family of geometries. A given microstructure family can be tested by performing simulations or experimental measurements on a set of samples drawn from it. The mapping between parameters and physical properties discovered in this testing process helps uncover the underlying design principles that drive these correspondences. In practical applications, mapping the parameter space also allows for the selection of a family member that has a desired tradeoff of physical properties [Gibson and Ashby, 1982]. Unfortunately, it is rare for manually designed microstructure families to reach extremal properties. This is because the space of possible microstructure designs is combinatorial and therefore impossible to explore exhaustively. One common approach to bypass this design challenge is to use computational methods, such as topology optimization [Sigmund, 1994, Vogiatzis et al., 2017], with a computer simulation in their inner loop to find a microstructure with a desired tradeoff of physical properties. Unfortunately, constructing parametric models from these optimized structures has heretofore required further expertise and manual design effort [Clausen et al., 2015]. In contrast to previous work, we present the first computational method to automatically explore the space of microstructure designs and discover parametric families optimized for competing properties.

While our methodology is not limited to specific physical properties, this study applied our method to design of mechanical microstructures. Specifically, we set our algorithm to search for a particularly interesting type of mechanical microstructures: auxetic materials, which have a negative Poisson's ratio. These materials have the unusual property of becoming laterally thinner under axial compression. 2D auxetic structures are well understood due to their relatively simple geometry such as reentrant structures [Lakes, 1987, Sigmund, 1994], chiral structures [Ha et al., 2016, Prall and Lakes, 1997] and rotating mechanisms [Babaee et al., 2013, Bückmann et al., 2014]. Generalizing existing 2D structures to 3D is challenging since a naive arrangement of 2D mechanisms often results in orthotropic or other anisotropic structures with low shear resistance. Such structures will prefer shearing deformation when the load is not aligned well with the auxetic direction. Additionally, since Poisson's ratio for orthotropic structures is unbounded, orthotropic auxetic structures are much easier to find than isotropic ones18. Lakes fabricated and tested the first isotropic 3D auxetic structure13. However, designing manufacturable 3D auxetic structures remains a challenging task due to its complexity. Only a handful of 3D design patterns have been fabricated and measured [Andreassen et al., 2014, Saxena et al., 2016]. This study led to the discovery of five families with negative Poisson's ratio and tunable shear resistance.

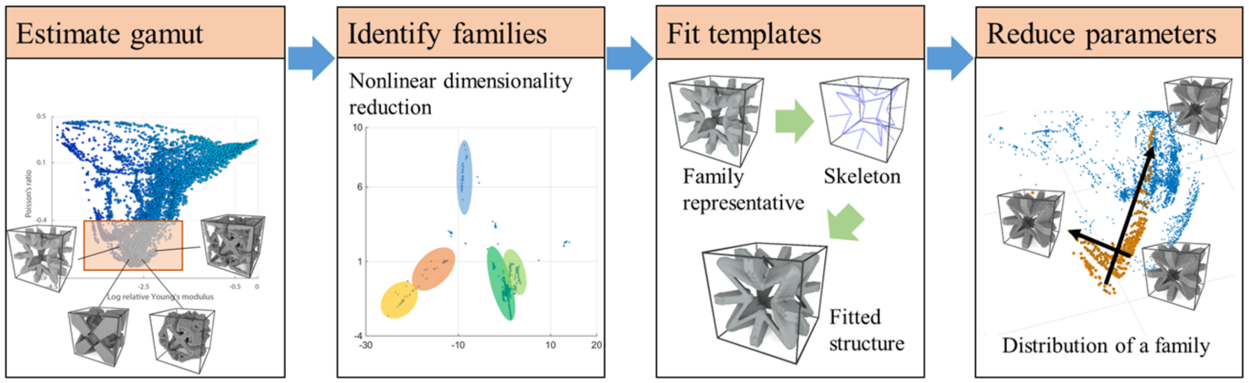


Figure 3-4: A computational process for discovery of extremal microstructure families. Given a set of physical properties and design constraints, we estimate the material property gamut using stochastic sampling and topology optimization. Structures near the gamut boundary are grouped into families using nonlinear dimensionality reduction. A representative from each family is fitted with a template represented as a skeleton. Beams are placed on the skeleton edges with optimized parameters to fit the original structure. Structure variations with the same topology can be generated by varying the beam parameters. Finally, reduced template parameters are computed to reveal domain-specific design principles.

3.3.1 Discovery Pipeline

Our discovery pipeline has four steps (Figure 3-4). The first step estimates the material property gamut, which is the range of material properties achievable by the microstructures. Here a microstructure is defined on a 3D regular grid composed of hexahedral voxels. The design space includes all possible material assignments to the voxels. Since exhaustively simulating all possible microstructures is impractical, this step computes a set of sample microstructures using methods outlined in Section 3.2. The topology optimization stage pushes structures past the explored gamut boundary along gradient directions. The stochastic stage introduces discrete changes to escape local optima.

In the second step, common geometric traits are identified among microstructures near the gamut boundary. Geometrically similar structures are grouped into families using nonlinear dimensionality reduction (NLDR). Isomap [Tenenbaum et al., 2000] is used as the reduction method because it can discover long sequences of related structures while keeping distant points separated. The effectiveness of NLDR depends on the distance metric that measures geometric difference. A smoothed Euclidean norm is chosen for robustness (Supplementary Fig. 1). NLDR outputs an embedding of the microstructures in a low-dimensional space where similar structures are closely packed. Microstructures in the embedding space are clustered using a Gaussian mixture model [McLachlan and Krishnan, 2007] where each cluster corresponds to a family. Families with a significant number (> 200) of members are extracted for further analysis.

The third step in our process constructs templates for each microstructure family (Figure 3-5). We observe that most of the extremal structures are composed from beams, plates and blocks. All of these structures can be represented as cuboids with different edge lengths. We therefore chose cuboids as the building blocks for microstructure templates. To find

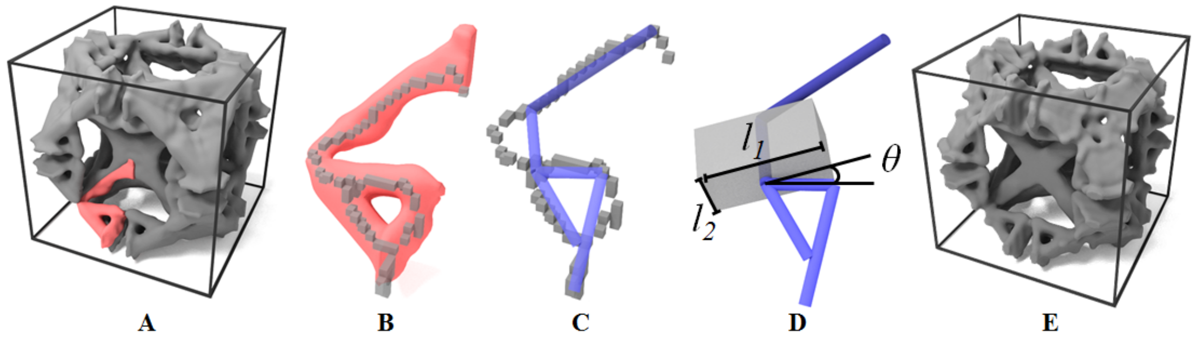


Figure 3-5: Steps for computing a microstructure template from a representative structure. For an input structure (a), we only need to analyze one tetrahedral slice (highlighted in red) of the whole structure due to its cubic symmetric construction. A morphological skeleton is extracted from the portion of the structure (b). The skeleton is a sequence of voxels. They are converted into a graph by connecting neighboring voxels. The graph is simplified (c) by merging paths into single edges while maintaining an error threshold. A cuboid (d) is placed on each edge of the simplified graph. (e) The final structure generated by the template using fitted parameters.

a template from a family representative, its topology is computed using a morphological skeleton [Lee et al., 1994]. The morphological skeleton is a subset of voxels in a 3D binary material structure that represents the branching and topology of the structure. From the skeleton, we construct a graph by connecting neighboring voxels. The graph is then simplified by collapsing paths into single edges. A path is a sequence of connected vertices where all intermediate vertices have a degree of 2. We then iteratively add back the furthest vertices to the simplified path until no vertex in the original path deviates away by more than a threshold of 0.02. The simplified graph is converted to a template by placing cuboid beams on each edge. To smooth the connections between the beams, we place dome-shaped caps at the endpoints of each beam. The cross-section sizing and orientation of each cuboid are initialized individually to minimize the Euclidean norm between the cuboid and the smoothed input structure. We then run gradient descent with central differencing to adjust all cuboid parameters including cuboid endpoint positions to arrive at a final fitted structure

The final step of our software pipeline computes reduced parameters to facilitate intuitive navigation in the material property space. Since the templates from the previous step contain tens of parameters that do not directly correspond to material properties, it is still difficult to understand the key design principles. The reduced parameters allow for direct tuning of each material property. For a given parametric template, its parameters are fitted to all structures of the corresponding family. To avoid outliers, microstructures leading to large fitting errors ($> 5\%$ voxel difference) are excluded. Principal component regression (PCR) is then performed on the set of fitted template parameters to find principal directions in the template parameters space. Varying the parameters in a direction corresponds to moving on the gamut boundary in a certain direction. A reduced parameter is assigned to each direction to control amount of change along that direction.

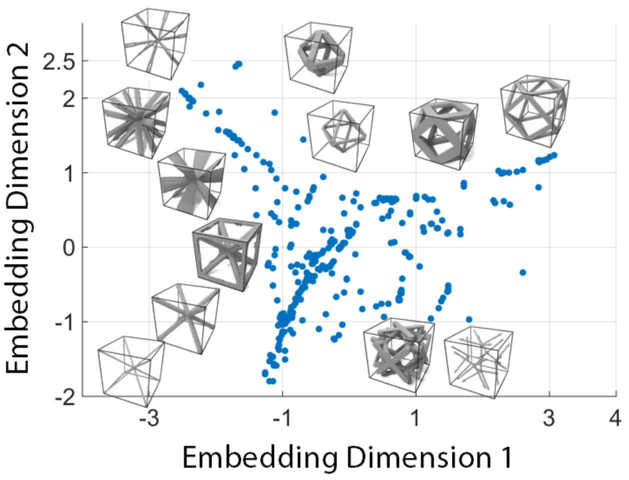


Figure 3-6: Structures with large Poisson's ratios ($\nu > 0.3$). The embedding is computed using Isomap with 3 dimensions while the plot shows the 2D projection of the embedding. Structures with a positive Poisson's ratio have relatively simple topologies. Most structures are controlled by a single beam reflected according to cubic symmetry.

3.3.2 Results and Discussion

The results of this study focus on elastic material properties: Young's modulus, Poisson's ratio and shear modulus. The elastic material property gamut is estimated from 15,000 3D cubic-symmetric microstructures at a voxel resolution of 64^3. The structures and material parameters are available online [Chen, 2017]. The voxel resolution is a power of 2 because that is necessary to achieve optimal performance of our multigrid FEM simulation. The specific resolution 64^3 is chosen because it is sufficient for discovering auxetic structures with a wide range of relative shear modulus while 32^3 structures cannot achieve comparable complexity or property ranges. The macroscopic elastic parameter of each microstructure is computed using homogenization theory [Guedes and Kikuchi, 1990, Xia and Breitkopf, 2015a] assuming a periodic boundary condition. Each microstructure consists of a per-voxel binary material assignment. Due to manufacturing limits on minimum feature size, sensitivity filter [Sigmund, 2007] is applied in gamut sampling step to avoid structures with overly thin features. Here we focus on analysis of auxetic structures since families with positive Poisson's ratios are relatively simple (Figure 3-6). Five families with significant number of members (Figure 3-7) are discovered using three Isomap embedding dimensions. We confirmed that Isomap associates seemingly distant structures through intermediate structures. For example, structure 5-1 and 5-3 from family 5 have very different beam thicknesses resulting in large geometric distance. However, the embedding reveals that there is a sequence of structures such as 5-2 that make the connection between them.

Three structures with different material properties from each family are printed to verify simulation accuracy. All structures are printed using an EOS SLS printer with elastic material PEBA2301. The printer required a minimum feature size of $0.9mm$ for wire diameters and $0.8mm$ for wall thicknesses. To satisfy the printing constraints, each cell is scaled to a side length of $2.54mm$. Simpler structures from Template 1-3 are printed using a 2x2x2 grid arrangement while more complex families are printed using a 3x3x1 arrangement to allow

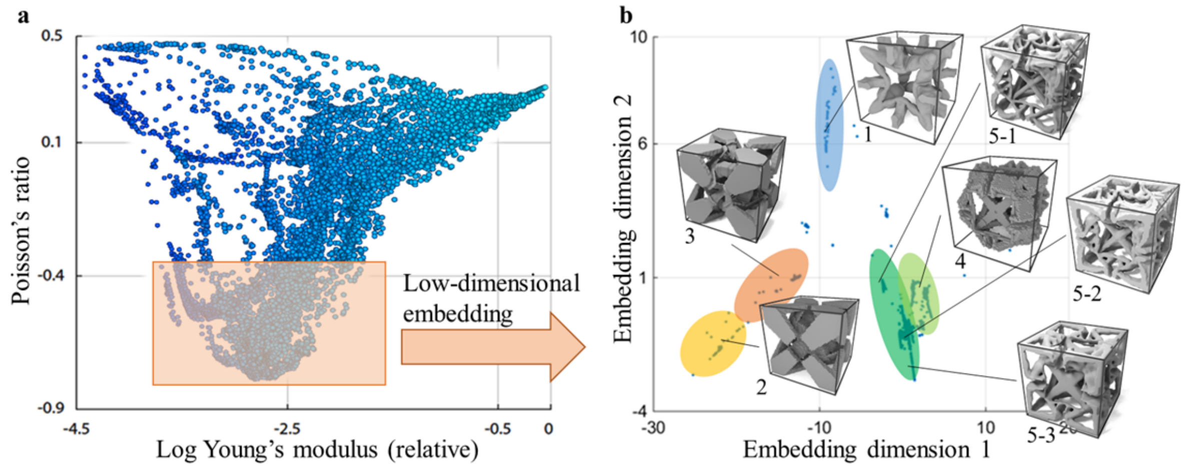


Figure 3-7: Five microstructure families identified by nonlinear dimensionality reduction (NLDR). Structures with similar properties in the gamut (a) are selected to study their commonalities. We focused on structures with negative Poisson's ratio (auxetics) since they exhibit more complex structures. Auxetic families are identified in the embedding space numbered from 1 to 5 (b). Families with similar topologies are located closer in the embedding space. Three example structures from family 5 show underlying connection between seemingly distinct structures through gradual morphing of shape.

support material to escape. The base printing material is measured using an Instron 5944 with tensile tests instead of compression tests since a solid block of base material is too stiff for our equipment. The PoissonâĂŹs ratio is measured using a video camera fixed on the test machine (Figure 3-8). Even though all samples are printed using the same printer and the same materials, the YoungâĂŹs modulus of the prints is highly variable. More specifically, the stiffest sample has a YoungâĂŹs modulus which is twice as high as the one of the softest samples. On the other hand, the PoissonâĂŹs ratio of the base print material is measured to be 0.34 and has a much lower variance of 0.02 in the sample set. The 3D structures are measured using a compression test at a speed of $2mm/min$ with $6mm$ maximum strain. The compression plates are lubricated with oil to reduce friction. Significant variance in PoissonâĂŹs ratio is observed due to several factors in manufacturing and measurement. An additional challenge is that the printer does not reliably reproduce the geometry specified by input files. In practice, the printed models are thickened by $0.1 - 0.4mm$, which is significant compared to the thinnest feature size in our microstructures. This stiffens the joints and reduces PoissonâĂŹs ratios. The effect is exacerbated by incomplete support removal. The support material is the same as the print material in powder form, which sticks to the print easily especially around hard-to-reach internal corners. We believe that the discrepancy can be reduced in the future by using more precise printing technologies with soluble support material. For each of the five families, a parametric template is automatically constructed. The initial topology of a template is extracted from the morphological skeleton of a representative structure (Figure 3-9b). While the topologies are visually complex, they are generated by mirroring a small number of beams (highlighted in red) reflected according to cubic symmetry (Supplementary Table 2). The most complex template 5 contains only

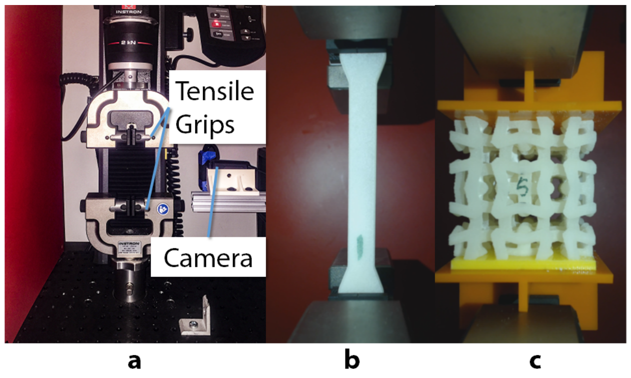


Figure 3-8: Test apparatus (a) for measuring YoungâĂŹs modulus and PoissonâĂŹs ratio using tensile (b) and compression (c) tests. The PoissonâĂŹs ratio is calculated using vertical displacements and horizontal displacements. The vertical displacement is read from both the tensile test machine and the camera for redundancy. The horizontal displacement is measured using the camera only.

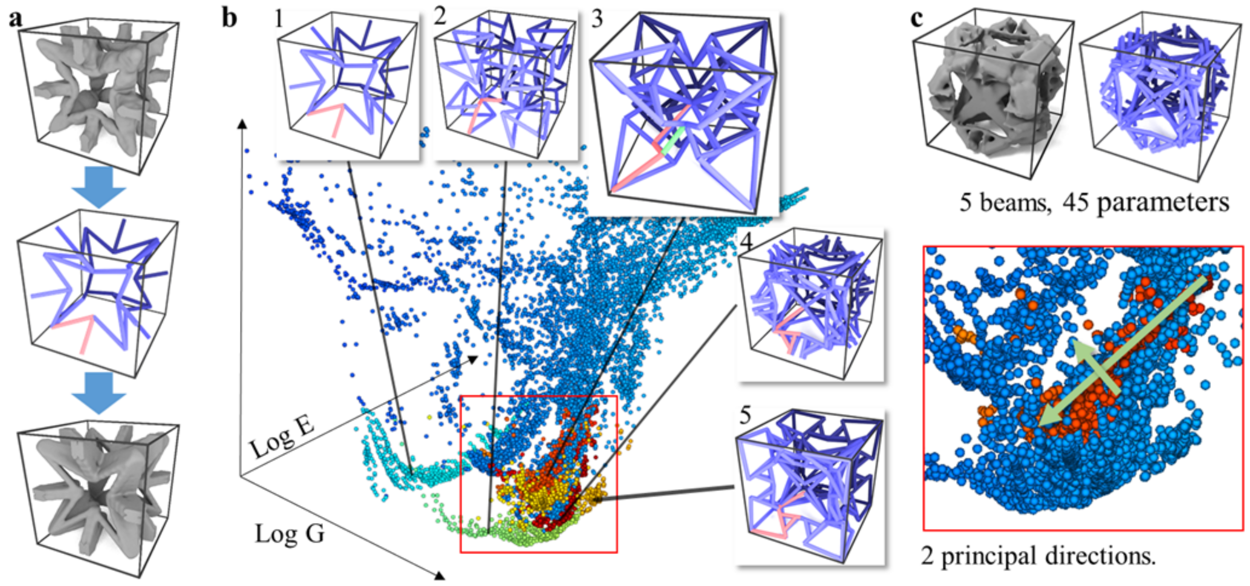


Figure 3-9: Sampled coverage of microstructure templates in the gamut. (a) Extracting a skeleton (middle) from a representative structure (top). The skeleton represents the topology of the structure. A beam network is derived from the skeleton by placing a cuboid on each edge of the skeleton. Since we enforce cubic symmetry, the beams in a single tetrahedron determine the entire beam network. A template can generate a new structure (bottom) that approximates the original structure. (b) Coverage of each template in the material property space. (c) Reducing template parameter dimensions with principal component regression. The first two reduced parameters approximately correspond to varying the Young's modulus and Poisson's ratio of a structure.

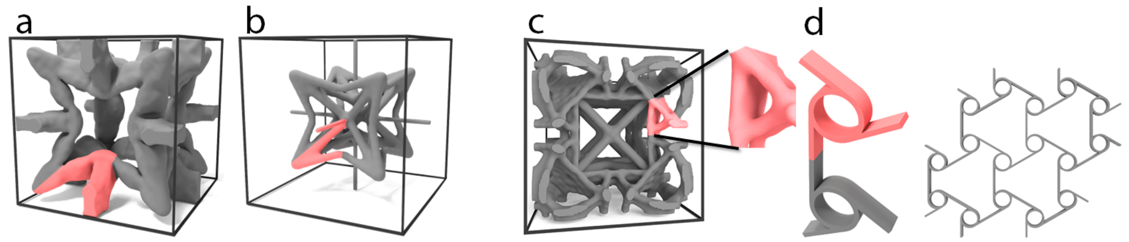


Figure 3-10: Microstructures that resemble designs from previous works. (a) A reentrant structure in our database similar to the conceptual sketch (b) proposed by Lakes [1987]. Both structures have very low shear modulus ratio (0.05-0.15). Our structure is simpler with only two control beams reflected by cubic symmetry while (b) has three beams (highlighted in red). (c) The rotating triangle mechanisms resembles 2D chiral structures [Prall and Lakes, 1997]. (d) An anti-trichiral lattice [Alderson et al., 2010] has unit nodes most similar to our rotating triangle joints (highlighted in red).

6 control beams. The five families cover similar ranges of Young's modulus and Poisson's ratio. However, they span different ranges of shear modulus. Inspired by classical linear elasticity theory, we compare the shear modulus ratio defined as

$$G' = \frac{2G(1+\nu)}{E},$$

where G is shear modulus, E is Young's modulus and ν is Poisson's ratio. For traditional isotropic materials, the theoretical ratio is one. A low ratio indicates low resistance to shear deformation. For auxetic materials, lower ratios are much easier to obtain than higher ones. Even with foam structures assumed to be isotropic, experimental data from previous work indicates that the ratio is less than one [Roh et al., 2013]. Template 1 resembles the conceptual sketch by Lakes [1987] and belongs to the reentrant class of geometry. The difference is that our template has only two beams mirrored by cubic symmetry while Lakes' sketch contains three (Figure 3-10). It is the simplest auxetic template that we identified, as our microstructure database does not contain any single-beam auxetic structure. The shear modulus ratio of this family falls in the range between 0.07 and 0.24, which is the lowest among all five families. Templates 2 and 3 are similar to each other and differ by a diagonal beam in the face center (highlighted in green in template 3). Since their geometric difference is small, they are adjacent in the Isomap embedding space. The central beam is responsible for increasing the shear modulus of the structures. For structures with ν around -0.5, the additional beam increases the maximum shear modulus ratio from 0.34 to 0.90. Templates 4 and 5 also differ by a single beam. Even the most complex template 5 is optimized from a simple cube frame through our continuous optimization step. The additional beam in template 5 makes the family stiffer overall compared to Template 4. Both families can achieve shear modulus ratio greater 1 for $\nu < -0.5$.

For each family, principal directions of template parameters are extracted using PCR. The templates and reduced parameters are included with the report. Two significant directions correspond to change in Young's modulus and shear modulus are kept for tuning structures. These directions reveal that for families 2 and 3, the thickness of the slanted column (Figure 3-

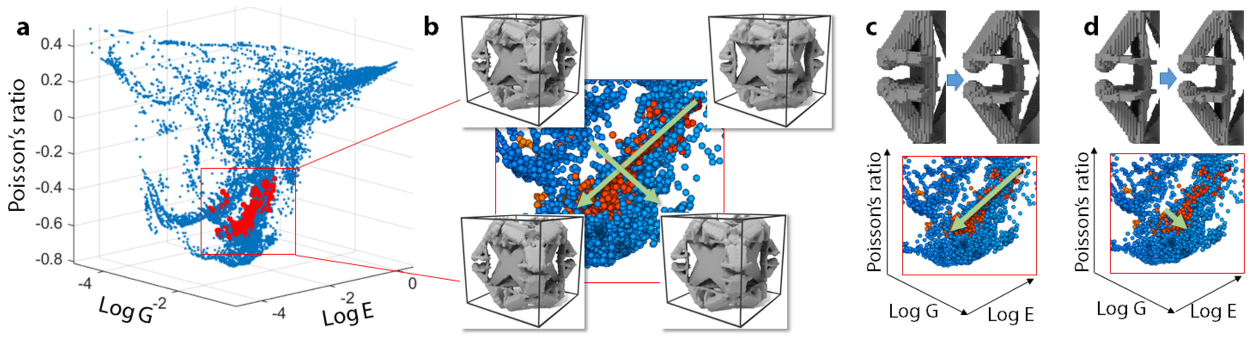


Figure 3-11: Reduced parameters for Family 4. The distribution of the parameters corresponding to Family 4 is shown as red points in (a). The two principal directions are shown as green arrows in (b). The first direction reduces the Young's modulus and Poisson's ratio by decreasing the joint thickness (c). The second direction increases the shear modulus by slightly rotating the triangle joints outward (d).

12a highlighted in red) is crucial for Poisson's ratio where the Poisson's ratio increases quickly with increasing beam thickness. For families 4 and 5, the thickness of the rotating triangle affects the tradeoff between Young's modulus and shear modulus (Figure 3-11). While our cuboid-based templates are very simple, they are sufficient for replicating the auxetic behavior of the corresponding families. We validated the auxetic properties of the fitted microstructures using simulation. New structures are generated by varying template parameters. 300 new structures are sampled from each family along two PCR coordinate directions. The coverage of the templates in the microstructure gamut (Figure 3-9b) shows that the templates can generate microstructures on the gamut boundary.

So far all of our simulations are carried out assuming linear elasticity, which is only accurate for infinitesimal deformations. We also make the common assumption that there is no self-collision. This assumption also imposes a limit on the maximum compressive strain we can apply to our structures before self-collision occurs. Representative structures from Families 4 and 5 have the lowest limit at 7% compressive strain. In practice, non-linear deformations such as bending and rotation are prevalent in our auxetic structures. Such deformations can cause linear elasticity to incorrectly predict significant volume expansion of rotated parts (up to 20% percent in our test cases). Thus, we tested our structures using a nonlinear material model to understand their behavior under large deformations. We simulated nonlinear deformation behavior using Neo-Hookean material model. At maximum allowed strain of 7%, linear elasticity and Neo-Hookean model still has acceptable agreement with an average error of 16% in computed Poisson's ratio. In addition to simulation, we also printed three example structures from each family with varying Young's modulus and material ratio. Our structures demonstrated consistent auxetic behavior (Supplementary Video 3) even though they are optimized with linear elasticity assumption. Our structures do not rely on structural instability [Bertoldi et al., 2010] for auxetic behavior and shrinks uniformly as load increases. This means that their deformations consistently follow the same pattern for different trials.

Our process automatically discovered two types of auxetic mechanisms: slanted columns and rotating triangles (Figure 3-12). The slanted column mechanism transforms vertical

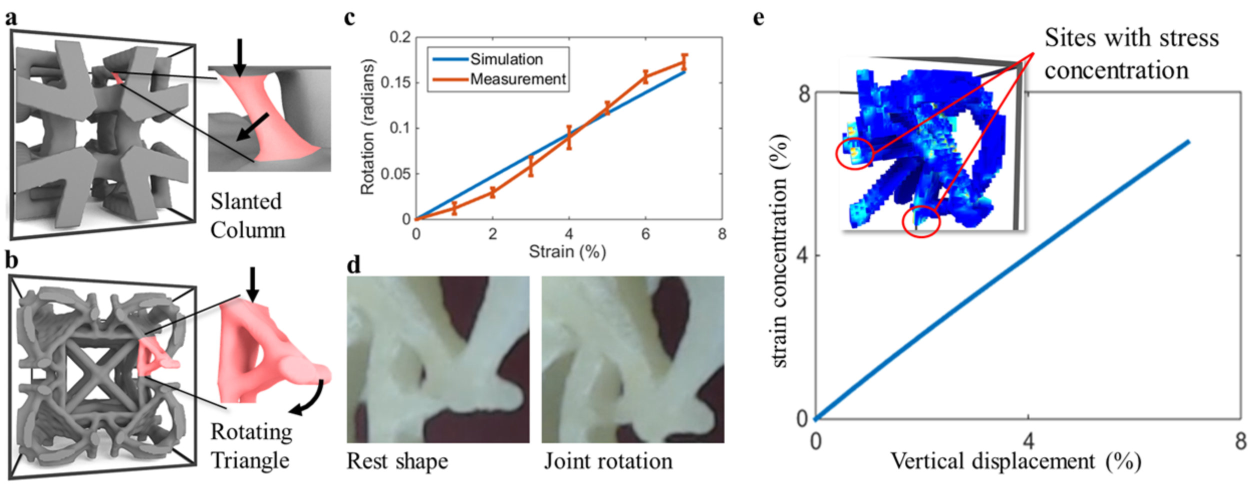


Figure 3-12: Discovered auxetic mechanisms. Two mechanisms capable of producing auxetic behavior are discovered from our microstructure families. The slanted column (a) transforms vertical stress into horizontal displacement. The rotating triangle mechanism (b) pulls the outer tip of the joint towards the center of the structure, reducing the macroscopic volume. (c) The relationship between vertical strain and rotation of the triangle joint. The rotation is observed in printed samples under vertical load (d). Stress is concentrated at the lower end of the triangle joint (e).

compression to horizontal motions. The rotating triangles transform vertical compression into a winding deformation that pulls the right end of the mechanism towards the center of the microstructure. Their motions are shown in supplementary video S3. While rotating triangles bear resemblance to existing 2D structures [Alderson et al., 2010] known as chiral structures (Supplementary Fig. 6d), its extension to 3D cubic structure with large shear modulus has never before been constructed. Additionally, the entire mechanism is discovered entirely automatically without imposing any artificial design restrictions—all microstructures are built from voxels. To inspire future applications of these mechanisms, we report the loading behavior of the mechanisms. These auxetic mechanisms are the most active parts in the microstructures. They act like joints that connect the more rigid scaffolding in microstructures. Because of this, they undergo the most deformation and concentrate a large amount of stress. For the rotating triangles, the stress is concentrated on the connections around the triangle. We computed the maximum principal strain in the structure with respect to the vertical compressive loading to provide insights into the strength of the block. At the maximal compressive loading (7%), the maximum principal strain in the structure is 7%. Calculation using a reported Young's modulus of $80MPa$ yields a von Mises stress of $6.72MPa$ (Fig. 4e) while our print material has a reported strength of $8.5MPa$. The printed structures are approaching the strength limit under the load. Since the available material is relatively weak even compared to common materials such as ABS plastics and rubber, we believe that structural strength can be improved significantly with future manufacturing materials.

We have shown a computational method that combines discrete sampling, continuous optimization and dimensionality reduction methods for automatic discovery of new microstructure families and mechanisms that would have been challenging to design manually. The discovered structures are suitable for manufacturing as they avoid thin features and distribute deformation over beams instead. They also span a wide range of shear moduli, allowing engineers to balance between different macroscopic properties. While our case study focuses on elastic material properties, the technique may be applied to other physical properties whenever predictive simulation exists. Our computational pipeline paves the way to discovery of structures that balance mechanical, thermal, optical, acoustic and electromagnetic properties. Moreover, it advances the understanding of underlying mechanisms that are crucial to extremal properties.

3.4 Topology Optimization

A classic topology optimization problem consists of optimizing the shape and structure of a given object defined by a prescribed domain in order to minimize a given cost function. For example, topology optimization can minimize the compliance of the object while satisfying the static equilibrium and the total weight constraint. Since the shape of the object is unknown *a priori*, only a design domain is specified as input. The design layout is voxelized and a density variable is assigned to every cell of the discretized domain. A high density corresponds to assigning material to a cell while a low density value implies a cell should be empty. By penalizing intermediate values for these densities, a binary distribution corresponding to the object's final shape and material can be eventually obtained.

In this work, we extend the traditional topology optimization algorithm in multiple ways. First, we do not compute a binary material distribution at the cell level as commonly done. Instead, we leverage our database of microstructures and ask for each cell to be filled with one of the microstructures. This is done by working with the macro-scale material properties of the microstructures instead of their geometry directly. The second difference is that our algorithm can be used with parametrizations of the material property space that are more complex than the single density parameter per cell that is commonly used in topology optimization algorithms. In our generalized formulation, each cell c_i contains an n-dimensional material parameter $\mathbf{p}_i \in \mathcal{R}^n$. We use $\mathbf{p}$ to denote the stacked vector of material parameters in all cells. Given a signed distance function $\Phi(\mathbf{p}_i)$ that defines the gamut, our new topology optimization problem is then written as

$$\begin{aligned} \min_{\mathbf{p}} : &\quad \mathcal{S}(\mathbf{p}, \mathbf{u}) \\ s.t. : &\quad \mathcal{F}(\mathbf{p}, \mathbf{u}) = 0 \\ : &\quad \Phi(\mathbf{p}_i) \le 0, \quad 1 \le i \le N_c \end{aligned} \tag{3.5}$$

where $\mathcal{S}$ is a real-valued objective function that depends on the material parameters and the displacement vector $\mathbf{u}$. $\mathbf{u}$ is an auxiliary vector for expressing elasticity equilibrium constraint and other displacement objectives. The equality constraint $\mathcal{F} = 0$ requires $\mathbf{u}$ to satisfy the elasticity equilibrium and the inequality constraint $\Phi \le 0$ guarantees that the material properties of each cell stay inside the precomputed gamut. The static equilibrium constraint is written as

$$\mathcal{F}(\mathbf{p}, \mathbf{u}) = K(\mathbf{p})\mathbf{u} - \mathbf{f_{ext}} = \mathbf{0}, \tag{3.6}$$

where $\mathbf{f_{ext}}$ are the external loads applied to the object and $K(\mathbf{p})$ is the stiffness matrix determined by the material parameters.

The gamut constraint for a point $\mathbf{p}_i$ in the material property space is described by an n-dimensional level set function $\Phi(\mathbf{p})$. We have $\Phi(\mathbf{p}_i) < 0$ for a point inside the gamut, $\Phi(\mathbf{p}_i) > 0$ for a point outside the gamut, and $\Phi(\mathbf{p}_i) = 0$ for a point on the boundary of the gamut. The value of Φ represents the n-dimension Euclidean distance to the level set boundary. The gradient of Φ are evaluated by a finite difference operation on the signed distance field.

In our examples, the material parameter $\mathbf{p}$ consists of the ratio ρ of the rigid material and the elasticity parameters $\mathbf{e}$. The objective function contains two types of terms: an elasticity term $\mathcal{C}(\mathbf{e}, \mathbf{u})$ that controls the deformation behavior (see Section 3.4.1) and an optional density term $\mathcal{V}(\rho)$ that controls the overall mass of the object. The density term is

$$\mathcal{V}(\rho) = (\sum_{i=1}^{N_c} \rho_i V_i - \hat{M})^2, \tag{3.7}$$

where V_i is the cell volume and $\hat{M}$ is the target overall mass. When one of the base material is void, the use of the density term allows to modify the topology of the object at a larger scale than the one of the microstructures, and thus to change the external shape of the object. In fact, even for multi-material designs involving base materials with similar mass densities, we noted that we could use the density term to encourage the presence of soft material in the structure. By removing the external cells entirely made of the soft material, we could then decrease the mass of the structure without significantly changing its mechanical behaviour. Alternatively, the density term can also be used to control other quantities related to the ratios of the different materials such as the cost of the object. For specific problems, we can also add spatially-varying weight control terms to Equation 3.7. For example, we can control the target weight of each individual cell by adding a local term $(\rho_i - \hat{\rho}_i)^2 V_i$.

We used a gradient-based numerical optimizer (Ipopt [Wächter and Biegler, 2006] in our implementation) to solve Equation 3.5. We enforced the elasticity equilibrium constraint using the adjoint method. The optimizer only needs to take the function values of $\mathcal{S}$ and Φ along with their gradients as input.

3.4.1 Elasticity Objectives

We used two types of objective functions for the elasticity term in our topology optimization algorithm. These two types of objectives allowed us to design a wide range of objects.

Target Deformation Our algorithm takes a vector of nodal target displacements and boundary conditions (external forces, fixed points, etc.) as input. Then, it automatically optimizes the material distribution over the object domain to achieve the desired linear deformation assuming a linear elastic behavior.

We define the deformation objective as

$$\mathcal{C}_d(\mathbf{e}, \mathbf{u}) = (\mathbf{u} - \hat{\mathbf{u}})^T \mathbf{D} (\mathbf{u} - \hat{\mathbf{u}}), \tag{3.8}$$

where $\hat{\mathbf{u}}$ is the vector of the target displacements, $\mathbf{D}$ is a diagonal matrix that determines the importance of each nodal displacement. We use $\mathbf{D}$ to define the subset of nodes that we are interested in. For example, we can set most entries of $\mathbf{D}$ to zero and focus on a portion of the domain (see Figure 3-20).

Topology optimization algorithms such as optimal criteria and MMA require the gradient of the objective function. The deformation objective can be differentiated using two simulations as follows. The first simulation computes the deformation of the design with the current material parameters.

$$\mathbf{u} = K(\mathbf{p})^{-1}\mathbf{f_{ext}}.$$

The second simulation computes

$$K(\mathbf{p})^{-1}(\mathbf{u} - \hat{\mathbf{u}}).$$

Putting these two equations together, we can compute the gradient with respect to material parameters.

$$\frac{\partial \mathcal{C}_d(\mathbf{e})}{\partial \mathbf{e}} = (\mathbf{u} - \hat{\mathbf{u}})^T K(\mathbf{p})^{-1}\mathbf{u}\frac{\partial K}{\partial \mathbf{p}}.$$

Minimum Compliance We have experimented with the same objective as the one used in the standard topology optimization algorithm where the compliance $\mathcal{C}_c$ is defined as

$$\mathcal{C}_c(\mathbf{e}, \mathbf{u}) = \mathbf{u}^T\mathbf{K}(\mathbf{e})\mathbf{u}. \tag{3.9}$$

The gradient of this objective is

$$\frac{\partial \mathcal{C}_c(\mathbf{e})}{\partial \mathbf{e}} = \mathbf{u}^T\frac{\partial K}{\partial \mathbf{e}}\mathbf{u}.$$

This gradient needs only one simulation to be computed.

In the commonly used SIMP formulation, the stiffness matrix $\mathbf{K}_i$ of each cell i depends on the artificial density value ρ_i through an analytical formula such as $\mathbf{K}_i = \rho_i^3\mathbf{K}_0$ where $\mathbf{K}_0$ corresponds to the stiffness matrix of the base material. In contrast, the stiffness matrix in our objective function is directly computed from the material parameters of the material space and forced to correspond to a realizable material thanks to our gamut constraints.

Like previous work, we regularized the problem to avoid checkerboard solutions by applying a smoothing kernel on the gradient to favor smooth variations of the material parameters over the object layout. Our optimizer supports multiple objectives by linearly combining weighted objective functions.

3.5 Mapping Material Properties to Microstructures

After running the topology optimization algorithm, we generate a printable result by replacing each cell in the object lattice by a microstructure whose material properties match the optimal ones. Material properties of the microstructures are computed using the homogenization theory which is more accurate with a smooth transition between the geometries of neighboring cells. While smoothness in the material parameters can be easily enforced, it

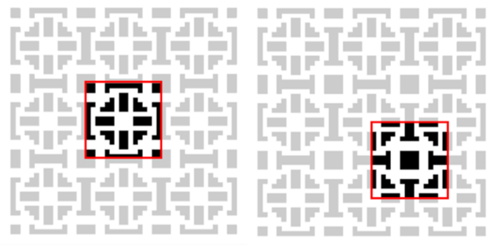


Figure 3-13: Under periodic boundary conditions, a microstructure pattern can be translated while preserving exactly the same material properties. These variants with the same properties givens us room to select structures while taking into account boundary connectivity.

does not imply topological similarity of nearby microstructures. For example, any translation of a given microstructure in a periodic tiling will result in a microstructure geometrically different but with exactly the same mechanical properties (Figure 3-13). Fortunately, our database is very dense and multiple microstructures generally map to similar points in the material property space, offering several variants. To further increase the number of possibilities, we also incorporate an additional exemplar for each microstructure by translating it by half its size, which preserves its cubic or orthotropic symmetry without changing its properties. We then run a simple but effective algorithm that picks the microstructure exemplars that minimize the boundary material mismatch across adjacent cells. We quantify this mismatch by

$$\mathcal{I} = \sum_{i=1}^{N_c} \mathcal{I}_i,$$

where $\mathcal{I}_i$ is the contribution associated to the cell i and corresponds to the number of boundary voxels filled with materials that are different from the ones of the voxels' immediate neighbours across the interfaces.

Our algorithm proceeds as follows:

- For each cell, select a list of possible candidates by picking all the microstructures with material properties in the vicinity of the optimal material parameter and initialize the cell with a random candidates.

- Compute the mismatch energy $\mathcal{I}_i$ associated to each cell i and sort the cells according to their energy.

- Pick the first cell in the sorted list, i.e. the one with the highest energy and assign to it the microstructure candidate that decreases the energy the most. If the cell energy does not decrease, move to the next cell in the list.

- Update the mismatch energies of all the impacted cells and we update the priority list.

- Repeat the last two steps until the mismatch energy $\mathcal{I}$ cannot be decreased anymore.

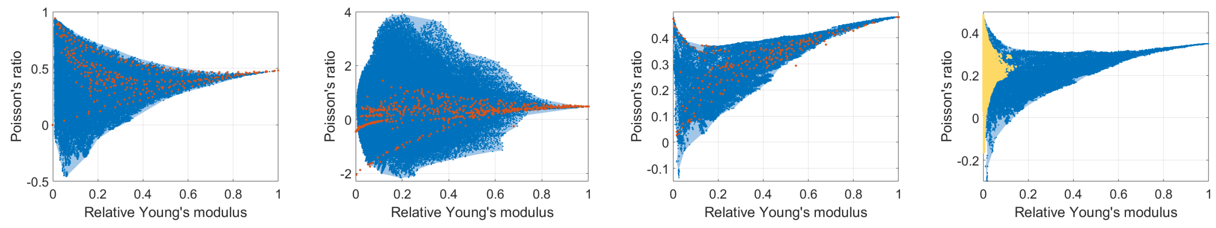


Figure 3-14: Gamuts computed with our discrete-continuous sampling scheme for 2D cubic structures (*left*), 2D orthotropic structures (*second from left*), 3D cubic structures (*second from right*) and 3D cubic structures with 0.35 as Poisson's ratio (*right*). The plots show the results for the projection of the gamuts on the plane defined by the macroscale Young's modulus along the x axis (normalized by the Young's modulus of the stiffest base material) and the Poisson's ratio corresponding to a contraction along the y-direction when the material is stretched along the x-direction. The blue dots correspond to the generated samples,the orange dots correspond to the microstructures from Schumacher et al. [2015] and the yellow dots correspond to the microstructures from Panetta et al. [2015].

3.6 Results and Discussion

We first analyzed our microstructure sampling algorithm for 2D and 3D microstructure gamuts. Then we used these precomputed gamuts to design and optimize a wide variety of objects with our topology optimization algorithm.

3.6.1 Microstructure Sampling

We evaluated our method on 2D and 3D microstructures made of one or two materials. For the 2D case we considered patterns with cubic and orthotropic symmetry that can be described by 4 parameters (3 elasticity parameters and density) and 5 parameters (4 elasticity parameters and density) respectively. In 3D we computed the gamut corresponding to cubic structures with 4 parameters. In all cases, the microstructure resolution is set to 16. We used two isotropic base materials with Young's modulus differed by a factor of 1000. They both have a Poisson's ratio of 0.45. We initially computed the databases for two-material microstructures, but also adapted these databases for microstructures made of a void and a stiff material. In the later case, we replaced the softer material by void, filter out all the microstructures with disconnected components and, in the 3D case, filled the enclosed voids and recomputed the homogenized properties. We provide a comparison between the initial and postprocessed databases in the supplementary material. The resulting gamuts are also depicted in Figure 3-14. Our databases contain 274k, 388k and 88k 2D cubic, 2D orthotropic and 3D cubic microstructures respectively and took from 15 hours to 93 hours to compute, which correspond to 68, 19 and 5 sampling cycles, respectively. We first compared our results to the ones obtained by Schumacher et al. [2015] and observed a significant increase in the coverage of the material space, even for 2D microstructures with a coarser discretization. This comforts us with the idea that topology optimization alone, while helpful to locally improve the microstructure geometries, is suboptimal for discovering the entire gamut of physical properties. The diversity of the microstructures that we obtained is also much

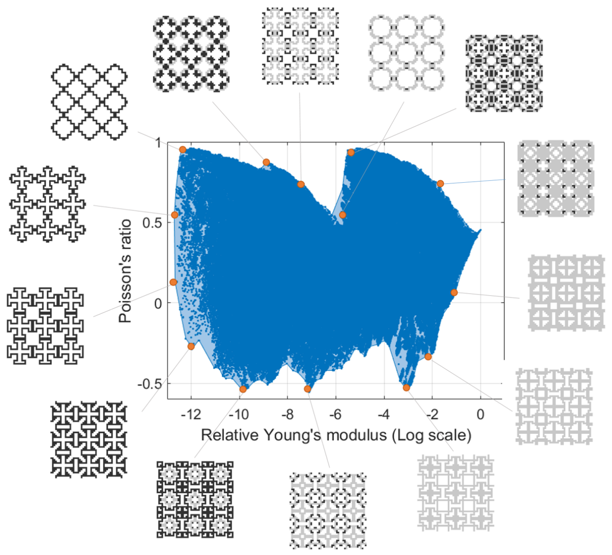


Figure 3-15: Gamut corresponding to 2D cubic microstructures made of two materials and void. The Young's modulus of the microstructures is plotted using a logarithmic scale. We show above some examples of microstructures lying near the estimated boundary of the gamut, i.e. with extreme material properties. The dark color corresponds to the softer material, while the light grey color is used for the stiffer material.

richer, thus providing a larger set of options for the practical use of microstructures. Note that they employed some regularization to avoid thin features. For 16^3 microstructures, we found regularization unnecessary since they are manifold and have a minimal feature size of 1/16 of the lattice size, which is the same order of magnitude as the thinnest parts of Schumacher's microstructures. For completeness, we also compared our database of 3D microstructures to the one of Panetta et al. [2015] at 16^3 and 64^3 grid resolutions (Figure 3-14, *right*). Our initial database was computed with 0.48 as Poisson's ratio. For this comparison, we then recomputed the material properties of the microstructures using the same Poisson's ratio as Panetta's at 0.35, which affects the extremal values of the obtained gamut. For the 64^3 microstructures, we used morphological operations in the discrete step and sensitivity filtering with a radius of 3 voxels in the continuous step to limit the minimum feature size to 1/32 of the lattice size [Sigmund, 2007]. Note that this comparison is provided for reference only since our microstructures are cubic while Panetta's are isotropic (a subset of cubic). Furthermore, they target a different 3D printing technology with self-supporting constraints not imposed here. Finally, we also obtained a dense sampling in the interior of the space, as a result of the randomness inherent to our approach. This reduces the need of running costly optimization in these areas and occurs even if we do not explicitly enforce any sampling there.

We also experimented with three-material 2D cubic microstructures. Two of the materials are solids with Young's moduli differing by a factor of 1000 and with 0.48 as Poisson's ratio, plus a void material. The resulting database contains about 800k microstructures that can potentially be printed. The corresponding gamut and some examples of the generated microstructures are shown in Figure 3-15.

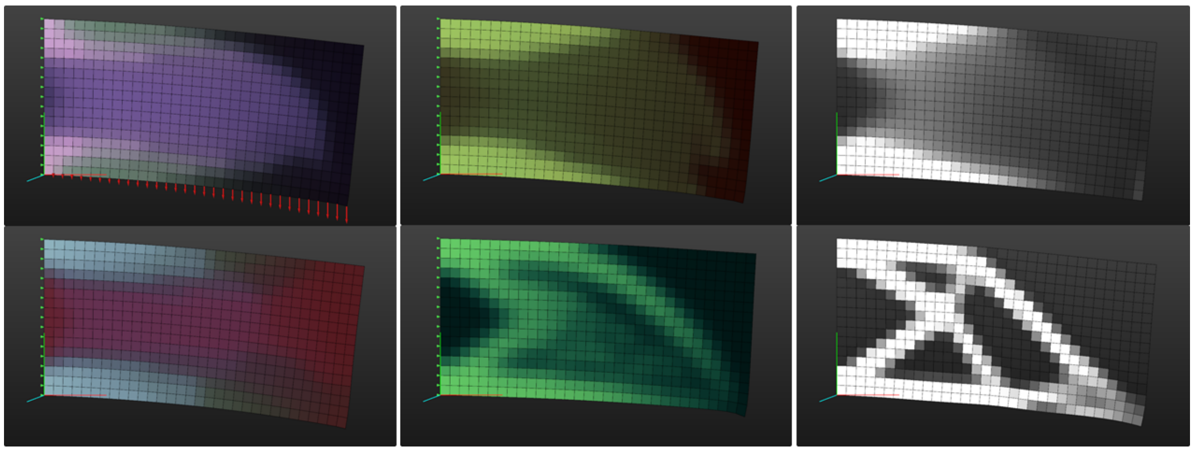


Figure 3-16: Material property distributions optimized in the orthotropic (*top left*), cubic (*bottom left*), isotropic (*top middle*) and an analytically defined gamut $E \geq \rho^3 E_0$ (*bottom middle*), with the material property space dimensions ranging from five to two. We compare our algorithm with the standard SIMP method with power index $p = 1$ (*top right*) and $p = 3$ (*bottom right*). For these figures, we computed the color of each cell by mapping every base vector of the normalized parameter space to a color range and linearly interpolating the colors associated to each of the parameters. In this example, the left side of the cantilever is fixed while a force distribution is applied to the bottom side (see red arrows in the top left picture).

3.6.2 Topology Optimization

We tested our topology optimization algorithm on a number of simple test cases and large scale examples. Detailed analysis and discussion of the results is provided below.

Impact of the Material Space We evaluated the impact of the chosen material space on a 2D cantilever beam with optimized minimum compliance. We tested our topology optimization algorithm with isotropic, cubic and orthotropic gamuts as well as the virtual materials used in the traditional SIMP approach. In the SIMP method, the stiffness of the material is $E = \rho^p E_0$, $p \geq 1$, where E_0 is the base stiffness and ρ is the material density variable. We also tested our algorithm on an analytical gamut with allowed stiffnesses E defined by $E \leq \rho^3 E_0$. The results are shown in Figure 3-16. It can be noted that, as the dimension of the material space increases, the final energy of the system decreases. This is to be expected since higher dimensional space means larger gamuts. Thus, when using cubic materials, the minimum compliance objective function reaches 3% lower energy than the standard SIMP method with power index 3. This difference reaches 11% when we use orthotropic materials. It is worth noting that the lowest elastic energy is achieved when we use the traditional SIMP method with $p = 1$ (as shown in Figure 3-17). However, this solution does not correspond to a realizable structure since some of the optimized materials do not correspond to any microstructure.

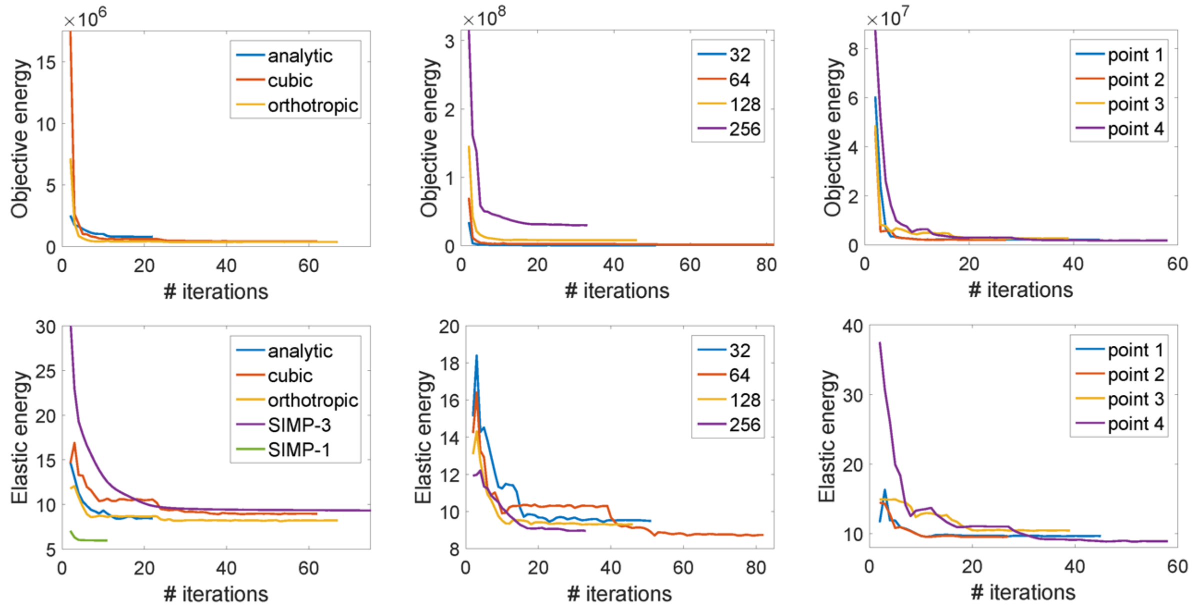


Figure 3-17: Convergence tests. Variation of the objective energy (*left*) and the elastic energy *right* of a beam being optimized for minimum compliance as the optimization progresses. The convergence plots correspond to the beam of Figure 3-16 when optimized using different material spaces (*top*), different resolutions for the beam lattice (*middle*) when using cubic microstructures, and different initial material properties for the cubic microstructures (*bottom*).

Matching Quality We evaluated for different examples the matching quality of the target deformation optimization. For the first test, we requested a beam to make an "S" shape when under tension (Figure 3-18). In order to avoid overfitting, we applied target displacements on the vertices of the boundary cells only. As depicted in the figure, the use of microstructures largely improves the global shape of the beam, which closely matches the target deformed shape. This becomes even more striking when compared to the behavior of a beam made of a homogeneous material. We also validated our algorithm by designing a soft ray whose wings can flap using a compliant mechanism (see Figure 3-19 and accompanying video). Boundary conditions are applied on two circular areas located along the spine of the ray. Each disk has one degree of freedom for deformation, namely contracting or expanding along the disk normals. This mechanism resembles one of many pneumatics-driven soft robots. We define two target deformation objectives corresponding to the flapping of the wings up and down, when alternatively contracting and expanding the two disks' boundaries. By running our multi-objective topology optimization framework, we can compute an optimized material design that can achieve both deformation modes when the corresponding boundary conditions are exerted.

Convergence and Robustness We evaluated the convergence rate of our topology optimization both on the minimum compliance problem and with the target deformation objective. For the minimum compliance problem, we used the same loading as the one of Figure 3-16. The corresponding results are shown in Figure 3-17 where we plot both the

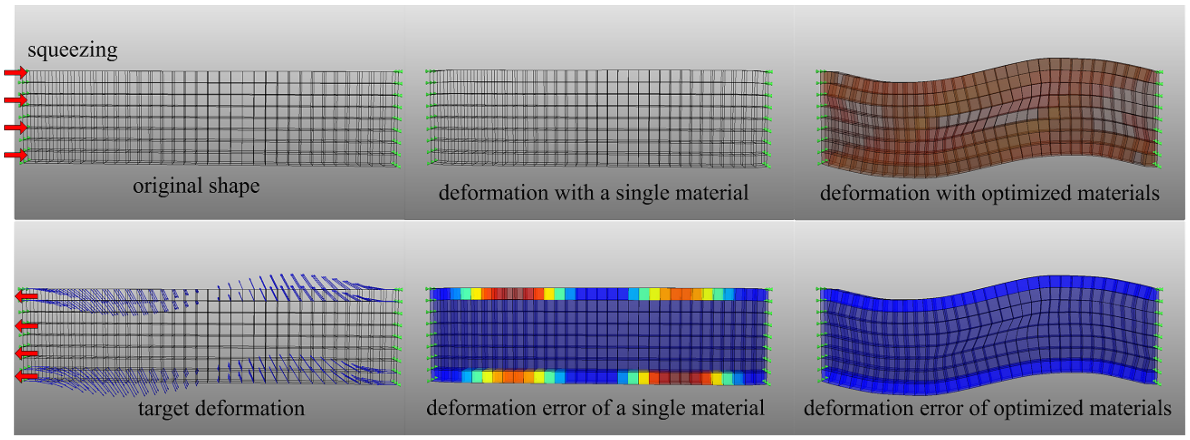

squeezing
original shape
deformation with a single material
deformation with optimized materials
target deformation
deformation error of a single material
deformation error of optimized materials

Figure 3-18: Optimizing a beam to take an "S" shape under compression (left column). A beam with homogeneous material can only compress uniformly (middle column). The optimized beam can deform as requested (right column). Target displacements are set on the horizontal boundary cells. The color plot for the bottom beams shows the deformation error of each cell defined by Equation 3.8.

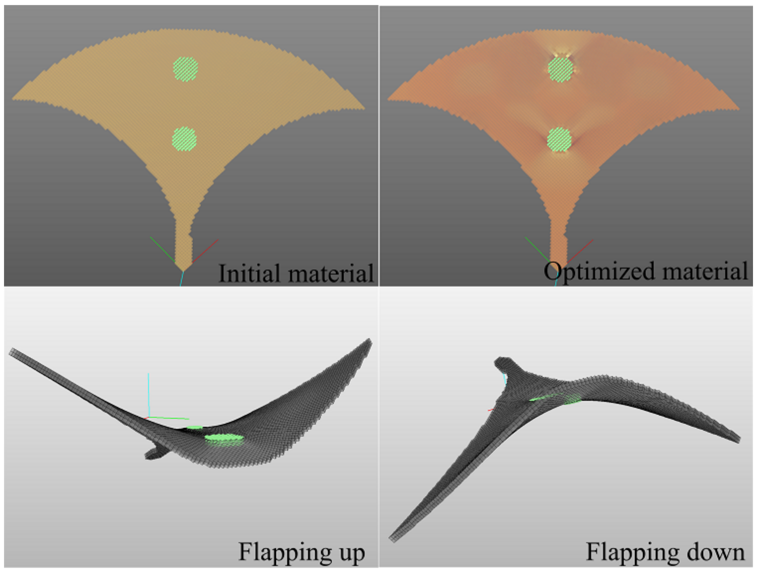


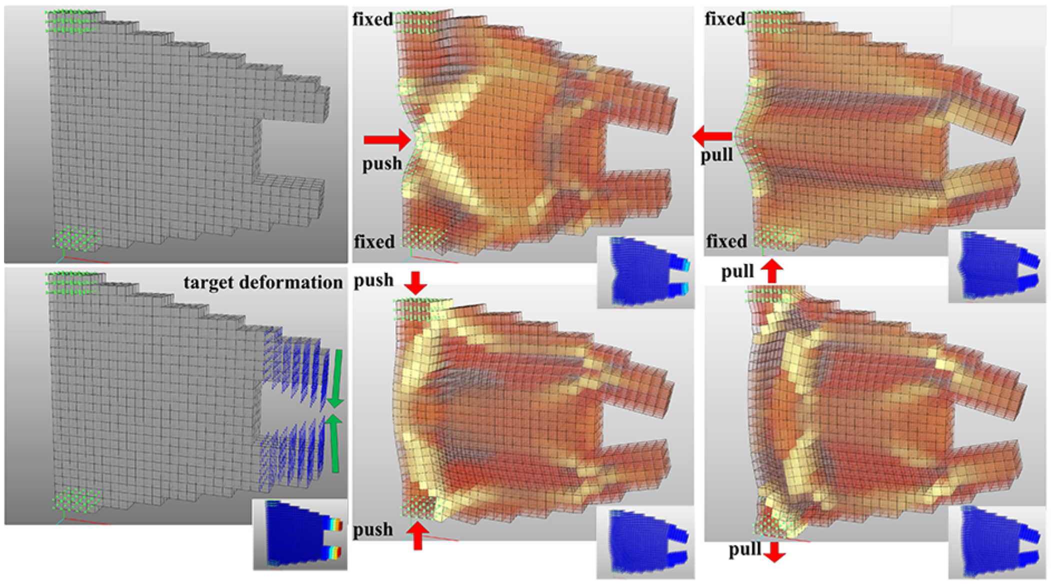
Figure 3-19: Designing a soft ray. The wings of the ray flap up and down when cells on its spine contract and expand. Constrained vertices are colored in green. The deformations achieved with the optimized materials are displayed on the bottom row.

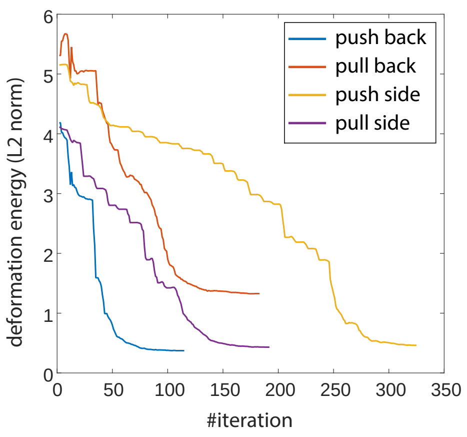
deformation energy of the structure as defined in Equation 3.9 and the original objective of the problem 3.5 that also includes the volume term defined by Equation 3.7. For all these examples, the algorithm converged after a couple of dozen iterations, irrespectively of the lattice resolution, i.e. the number of variables and the number of non-linear constraints. This demonstrates the scalability of the our algorithm. We also tested the robustness of our algorithm by starting with different initial conditions. In this case, we initialized the material parameters of each cell with a random material point projected onto the boundary of the gamut. Similar to other topology optimization schemes, we have no guarantee that we reach the global minimum of the function, and indeed, our algorithm sometimes converges to different solutions. However we note that these different solutions have a similar final objective value and are therefore equally good.

For the evaluation of the target deformation optimization, we tested the convergence rate when optimizing for functional mechanisms. To this end, we designed several grippers that can grasp objects by moving their tips when external forces are applied to their extremities. We experimented with four sets of boundary conditions, namely, pulling and pushing the back of the gripper horizontally, and compressing and stretching the extremities of the gripper vertically. As shown in Figure 3-20, these different settings lead to different material structures. The deformation errors of all the four designs converge to a low level after a couple of hundreds of iterations.

Accuracy We evaluated the accuracy of our algorithm on several optimized structures by comparing the deformation obtained when using the optimized homogenized material properties for each cell to the one obtained by a high resolution simulation in which every cell is replaced by its mapped microstructure. We first evaluated the accuracy on a deformable bar, one side of which was rigidly attached while the other was subject to different sets of external conditions (see Figure 3-21). We used a $8 \times 2 \times 2$ lattice to represent the bar with homogenized cells, which translates into a $128 \times 32 \times 32$ grid for the full resolution mesh. Similar stretching, bending and shearing behaviors were obtained for both sets of models. From a quantitative point of view, the differences amount to 5-10% in terms of average vertex displacement and $9\% - 33\%$ in terms of elastic deformation energy (see Table 3.1). We further evaluated the effects of material patterns by running a similar comparison on a cube made of periodic layers of similar microstructures and with random assignments of microstructures (Figure 3-22). As reported in Table 3.1, we show that the ratio between the magnitudes of the average vertex displacement differences is between 4% and 7%, and the elastic energy difference is between 10% and 19%.

Finally, we also compared the behaviors of one of the grippers (Figure 3-24). The original optimized gripper is made of 3k elements while the high resolution version is made of 4M voxels. Overall, the two models exhibit similar global deformation behaviors, in particular in the tip area. Some differences can be observed on the left side of the gripper for which the high-resolution model exhibits a lower effective material stiffness than its homogenized counterpart. With the same displacement boundary conditions applied, the high-resolution model deforms about 25% more than the homogenized model. The differences observed between the homogenized model behaviour and the full resolution simulation can be explained by two major factors: (i) numeral stiffness when using larger elements which tends to make





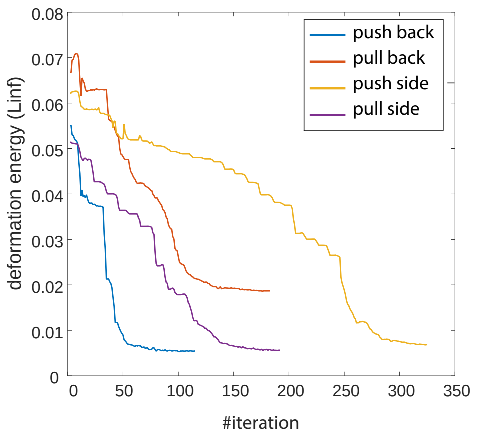


Figure 3-20: Designing functional grippers. The left column shows the rest shape of the gripper and the target deformation for the tip. The green dots correspond to the fixed vertices while the blue arrows are the target displacements. The middle and right columns correspond to the optimized results obtained for the specified boundary conditions. The inset pictures color-code the initial and final deformation error for the different examples. The convergence plots in the bottom row depict the change in the sum of the deformation errors corresponding to all the cells (left) and the value of the maximal cell error contribution (right) as the optimization progresses.

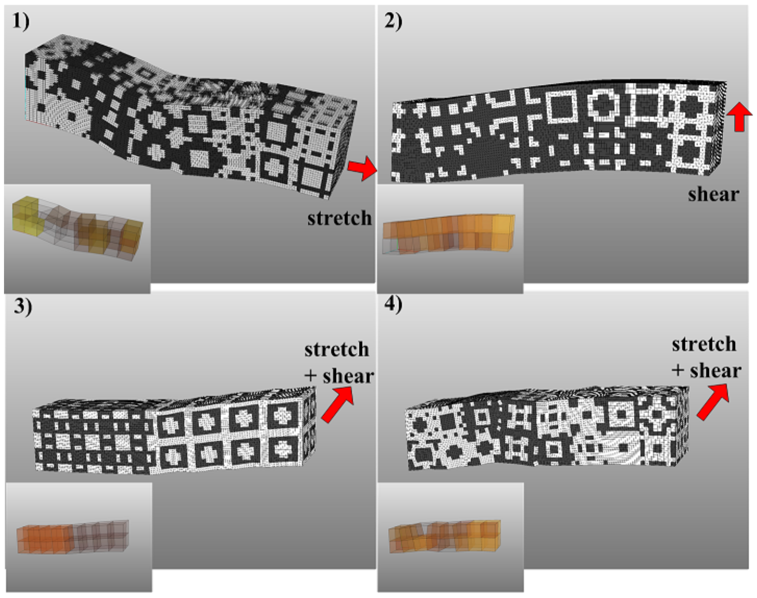


Figure 3-21: Comparison of simulated beams with homogenized material properties (*inset pictures*) to the ones using full microstructures (*large pictures*).

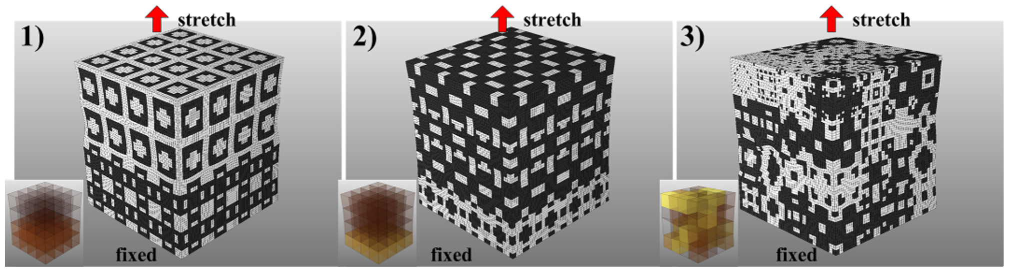


Figure 3-22: Comparison of simulated cubes with different material patterns modeled by homogenized cells (*inset pictures*) and full resolution microstructures (*large pictures*).

the homogenized mesh slightly stiffer in particular when bending deformation arises, (ii) violation of the periodicity assumption when replacing each cell by a single microstructure. This issue can be reduced by replacing each cell by a *tiling* of microstructures. This was verified on a cube made of a periodic arrangement of a single microstructure (see Figure 3-23). And indeed, as we increase the resolution of the simulation grid, the error between the homogenized model and the full resolution version decreases and converges to similar values.

Orthotropic materials We tested the behavior of our algorithm in a 5-dimensional space by using the gamut of 2D orthotropic microstructures depicted in Figure 3-14 (*middle*). To this end, we used a regular lattice whose vertices on the left side where fixed and we applied parallel forces on the vertices of the opposite side. The goal in the test was to minimize the compliance of the structure. As can be seen in Figure 3-25 and in the accompanying video, we experimented with different force directions. Unsurprisingly, when a single cell is considered, the microstructure that we obtain has a structure that is aligned with the direction of the

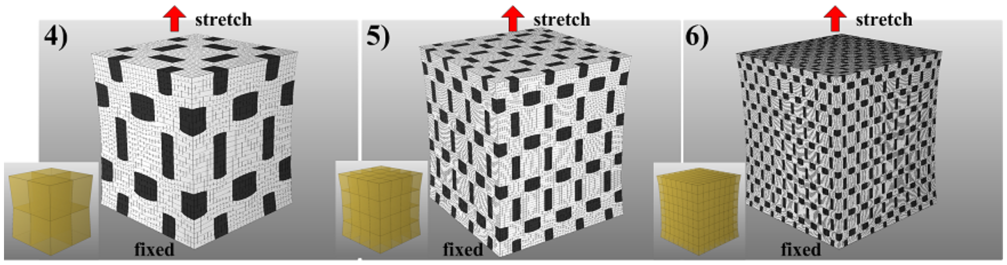


Figure 3-23: Simulation of a cube made of a periodic arrangement of a single microstructure at different resolutions. Inset pictures correspond to the model with homogenized material properties, while main pictures correspond to the full resolution simulations.

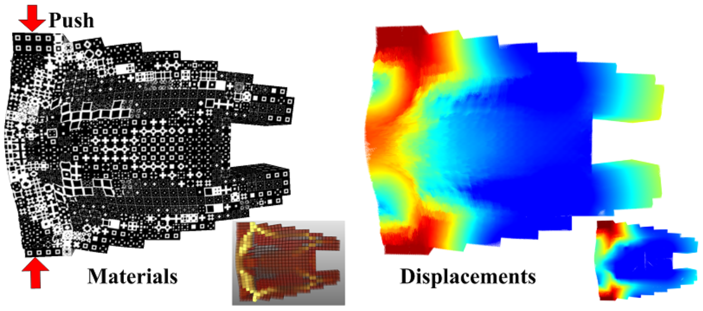


Figure 3-24: Comparison of a simulated gripper with homogenized material properties (*inset picture, left*) to the one using full microstructures (*main picture, left*). The figures on the right show the vector field of vertex displacement of the two models. The blue-to-red colors represent the magnitudes of the displacements.

Table 3.1: Error of simulation using homogenized materials (SI units). The simulated shapes are shown in Figure 3-21, 3-22, 3-23 and 3-24 The size of one microstructure is set to $1\times1\times1$.

Example	Mean displacement	Mean displacement difference	Elastic energy homogenized	Elastic energy full resolution
Beam 1	6.47×10^{-3}	6.04×10^{-4}	6.85×10^{-5}	6.17×10^{-5}
Beam 2	6.47×10^{-3}	6.04×10^{-4}	1.63×10^{-5}	1.08×10^{-5}
Beam 3	5.07×10^{-3}	4.97×10^{-4}	2.38×10^{-4}	2.07×10^{-4}
Beam 4	8.78×10^{-3}	4.45×10^{-4}	3.33×10^{-4}	2.30×10^{-4}
Cube 1	3.62×10^{-3}	2.86×10^{-4}	3.64×10^{-3}	3.20×10^{-3}
Cube 2	4.35×10^{-3}	1.94×10^{-4}	6.82×10^{-3}	5.94×10^{-3}
Cube 3	5.42×10^{-3}	4.22×10^{-4}	7.81×10^{-3}	6.32×10^{-3}
Cube 4	5.22×10^{-3}	4.89×10^{-4}	2.08×10^{-2}	1.63×10^{-2}
Cube 5	5.21×10^{-3}	2.17×10^{-4}	1.89×10^{-2}	1.63×10^{-2}
Cube 6	5.21×10^{-3}	1.32×10^{-4}	1.82×10^{-2}	1.63×10^{-2}
Gripper	1.32×10^{-2}	6.90×10^{-3}	8.67×10^{-3}	5.70×10^{-3}

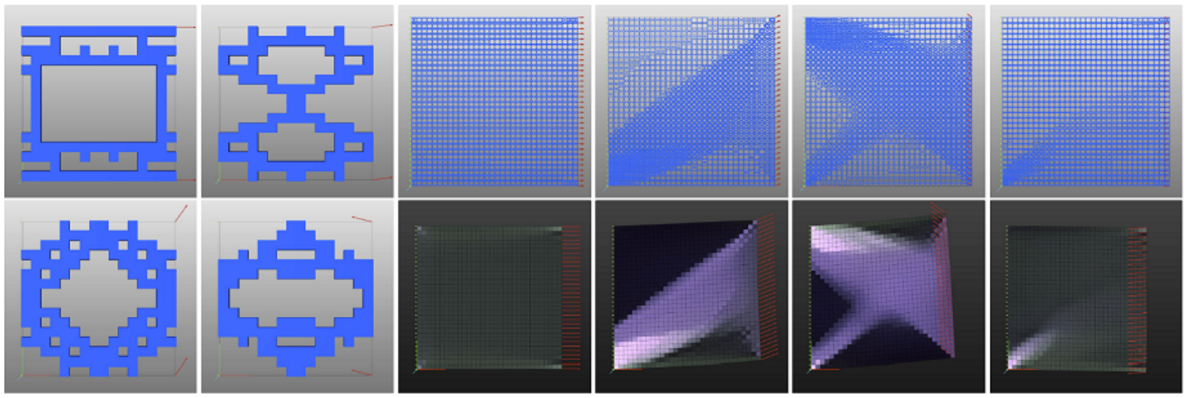


Figure 3-25: Optimizing the orthotropic material parameters of a single cell (*left*) and a 32×32 lattice of cells (*right*) subject to directional forces. The vertices on the left side of the layout are fixed while forces are applied on the right vertices as depicted by the red arrows. Our simple but effective tiling algorithm allows to nicely transition between microstructures of smoothly material properties (*right, top*).

forces (see Figure 3-25, *left*). For a higher resolution lattice this is no longer true and the resulting overall structure becomes less intuitive (see Figure 3-25, *right*). Note that the resulting material distribution varies smoothly. By considering various alternative for each material point, our tiling algorithm is able to map the material properties to microstructures which are well connected.

3.6.3 3D-Printed Designs

Leveraging our two-scale approach, we used our topology optimization algorithm to generate a wide variety of high resolution models that we 3D-printed. We used a Stratasys Objet Connex 500 and the two base materials *Vero Clear* and *Tango Black Plus* and used the database containing the three-dimensional cubic microstructures. The sizes and computation times of the resulting models are outlined in Table 3.2.

Since Ipopt performs a line search at each gradient step, one single step may correspond to multiple simulations. We show the average time required for taking a step in the last column of Table 3.2. For these large scale examples, Ipopt takes two hundred iterations in average to find a local minimum. Since our problem is formulated as a very general constrained continuous optimization, it is independent of the optimization package that is used and its speed could potentially be further improved by using alternative minimizers. We found Ipopt to be a good choice for its capability to efficiently handle a large number of inequality constraints, which is not the case of other popular minimizers used in topology optimization such as the method of moving asymptotes (MMA).

Our algorithm is mainly directed towards engineering applications and targets the design of objects undergoing small deformations. In the following examples, we sometimes intentionally exaggerated the target displacements (and scaled the external forces accordingly) for better visualization, which does not change the output of the algorithm with a linear material model.

Table 3.2: Statistics on the 3D-printed models. The last row uses the database of 64^3 microstructures.

Example	Grid Size	# Voxels	Time per FEM Solve [s]	Time per Step [s]
Beam	96×24×4	38M	0.7	5
Flexure	32×32×16	67M	1	12
Gripper	64×32×8	67M	1.7	10
Bridge	128×64×32	1074M	27	81
Bridge 2	320×160×80	1074G	1.3k	-

Beams with controlled deformation behaviour We started by designing a 3D hollowed beam with a desired deformed shape. The beam was stretched by moving vertices on two opposite sides. Our topology optimization algorithm was run using a target deformation objective. The resulting optimized material properties and the 3D-printed structure are depicted in Figure 3-26.

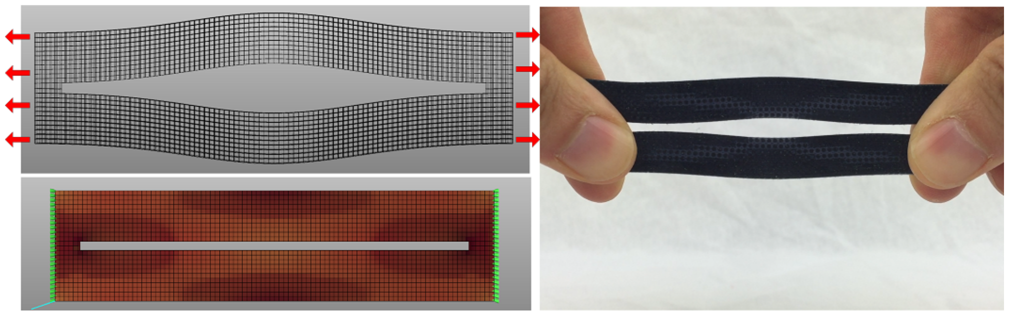


Figure 3-26: An optimized hollow beam with target deformation. The left figure shows the target deformation and optimized material distribution. The right figure shows the 3D-printed structure and the achieved deformation.

Multi-Objective Flexure Design We tested our algorithm on a multi-target deformation setting by optimizing the structure of a flexure mount with two different target shapes (see Figure 3-27). Here, our goal is to design a flexure that resists vertical loads while remaining compliant to horizontal loads. We assume that the object mounted on the flexure is connected to the flexure using a cylindrical connector that transmits the forces to the flexure via the connecting area. In the first scenario, vertical forces are applied to the points of the cylindrical area and we ask the flexure to stay as close as possible to its rest configuration. In the second scenario, horizontal forces are applied to the points of the cylinder and we ask the flexure shape to match the shape shown in the Figure 3-27.

Gripper We verified the functionality of our grippers by fabricating two of them. For these results we ran the optimization on high resolution meshes of the version that grasps the object when the extremities of the gripper are pressed (see Figure 3-28). By changing the parameter controlling the ratio of the soft material, different designs based on different mechanisms can be achieved. When more soft material is used the gripper achieves its target

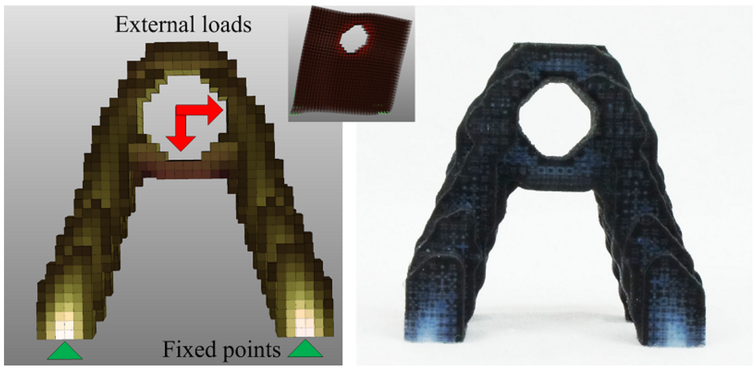


Figure 3-27: Optimizing a flexure mount. The flexure is connected to an object thanks to a cylindrical connector. We leave space for this connector by keeping a cylindrical area of the design layout empty of material. The material distribution of the flexure is optimized for two sets of external forces applied to the cylindrical area. Under vertical load, the flexure should stay close to the rest shape while under horizontal load, the flexure should deform according to the inset figure.

deformation thanks to out-of-plane bending, while for stiffer designs, the grasping motion is achieved via in-plane deformation.

Minimal Compliance Examples To demonstrate the scalability of our algorithm, we designed two bridges of increasing resolutions. The first bridge was optimized using a lattice of half a million cells which corresponds to 1 billion voxels (Figure 3-29). For the second bridge, we used the database of 64^3 microstructures and a layout made of 4 million cells, which amounts to 1 trillion voxels. We initialized the topology optimization by running the algorithm on a lower resolution grid with 1.4 million elements and used the resulting parameters as initial material parameter values for the higher resolution optimization.

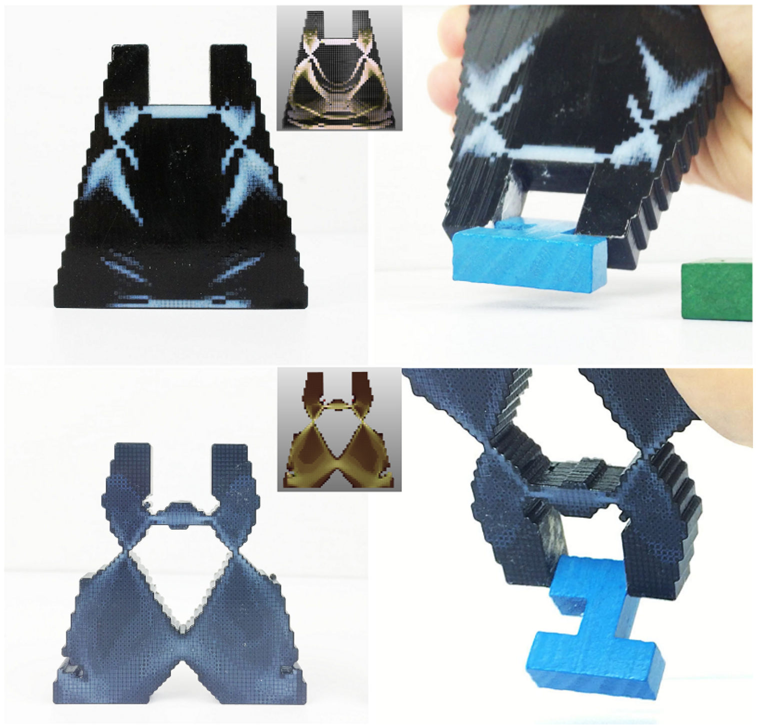


Figure 3-28: 3D-printed functional grippers. By setting different target ratios of the rigid material, different designs can be obtained. When more soft material is used the grasping behavior of the gripper is obtained via out-of-plane bending (*top*), whereas more rigid material is used, the gripper deformation remains planar (*bottom*).

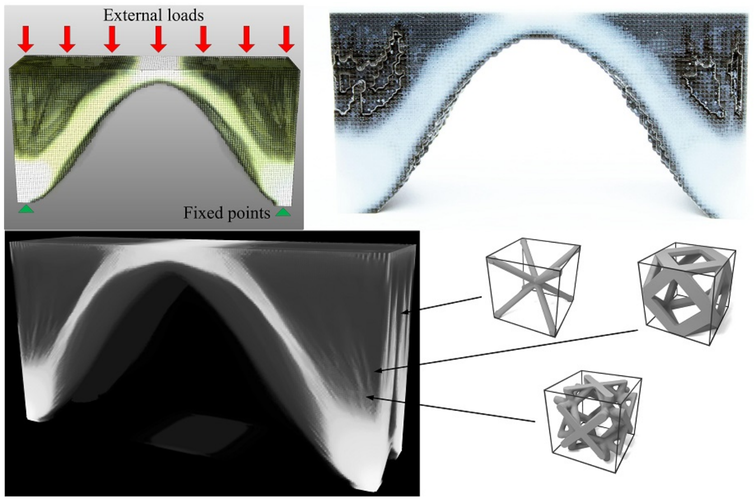


Figure 3-29: Optimizing a bridge. The initial layout corresponds to a 128x64x32 regular grid. We apply uniform loads on the upper plane deck. We compute the material parameters and set cells with extremely low stiffness to void (*top left*). We look up the microstructures and 3D print the bridge (*top right*). We scaled the problem to 1 trillion voxels by using a lattice of 4 million elements where each element corresponds to a 64^3 microstructure (bottom).

Chapter 4

Designing Dynamic Mechanisms

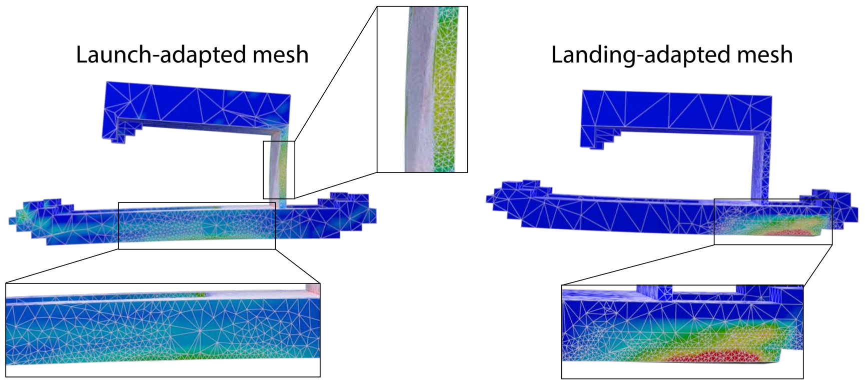
The realistic simulation of highly-dynamic elastic objects is important for a broad range of applications in computer graphics, engineering and computational fabrication. However, whether simulating flipping toys, jumping robots, prosthetics or quickly moving creatures, performing such simulations in the presence of contact, impact and friction is both time consuming and inaccurate. In this paper we present Dynamics-Aware Coarsening (DAC) and the Boundary Balanced Impact (BBI) model which allow for the accurate simulation of dynamic, elastic objects undergoing both large scale deformation and frictional contact, at rates up to 79 times faster than state-of-the-art methods. DAC and BBI produce simulations that are accurate and fast enough to be used (for the first time) for the computational design of 3D-printable compliant dynamic mechanisms. Thus we demonstrate the efficacy of DAC and BBI by designing and fabricating mechanisms which flip, throw and jump over and onto obstacles as requested.

4.1 Introduction

We present a pair of new methods to accurately simulate geometric and material nonlinearities subject to frictional contact, large loads and high-speed collisions at rates significantly more than an order-of-magnitude faster than previously available. Our methods combine efficiency and accuracy to enable design-for-fabrication optimization. They can be used for both fast, realistic animation and engineering analysis.

Here we look towards a new generation of efficient mechanisms for practical *dynamic* function [Lipson, 2014, Reis, 2015, Reis et al., 2015, Rus and Tolley, 2015]. In order to extend physics-driven computational design to this domain, however, a bottleneck must be overcome - the physical simulation itself. Simulations must accurately replicate the behavior of elastic materials subject to high-speed, transient dynamics. Modeling these systems combines many of the remaining grand challenges in simulating elastica. Specifically we must accurately resolve nonlinear elasticity, large deformations, stiff materials, high-speed dynamics, rapid loading and unloading, frictional contact, internal friction, high-speed collisions, and rebound.

State-of-the-art FEM systems currently able to accurately match these effects are exceedingly expensive - runtimes on the order of days are standard to perform a single simulation



Simulation mesh	Remeshing time(s)	Tetrahedra	Nodes
Launch-adapted	1.550	150,192	222,109
Landing-adapted	0.338	31,261	51,999
DAC-coarsened	none	5,488	7,898

Figure 4-1: Comparing adaptivity and coarsening. A tetrahedral model of a jumper is adaptively remeshed to capture input stress fields from its launch and landing states. We compare the resulting element and node counts to our DAC-coarsened model. We set the minimum edge length for remeshing to one that was experimentally found to yield convergent numerical results.

in many cases [Belytschko et al., 2013]. Thus, while generating a single simulation for visualization or animation is already time consuming, the many simulations required during design optimization compound an already prohibitive computational burden.

4.1.1 Efficiency with Accuracy

Let us explore four potential solutions for constructing fast *and* accurate simulation algorithms: (1) higher-order elements, (2) adaptive meshes, (3) reduced models, and (4) numerical coarsening. Can these methods provide the necessary efficiency to enable design-optimization while obtaining the predictive accuracy required to match fabricated results?

Higher-order elements offer us the opportunity to replace thin regions of our models with elements that capture higher-order deformation modes. While an attractive strategy, this poses two challenges. First, due to changing design parameters, we will need to identify suitable regions on-the-fly in order to perform this replacement. Second, coupling swapped-in higher-order elements to other element types introduces overhead. Consider, for example, replacing lower-order hexahedra with plate elements in these regions. We must then ensure continuity of displacements between plate-like portions of the design and thicker, volumetric portions. This, in turn, requires introducing difficult coupling constraints Bergou et al. [2007], Martin et al. [2010]. Finally, even with such additional efforts, these substitutions may not always improve computational performance as high-order elements contain more DoFs than their low-order counterparts Belytschko et al. [2013].

Adaptive meshing allows us to reduce element counts in material regions where refinement is less critical. Let us ignore the difficulty of implementing adaptive meshing and the per time step cost to remesh. Even so, adaptive meshes are challenging in our setting. We model

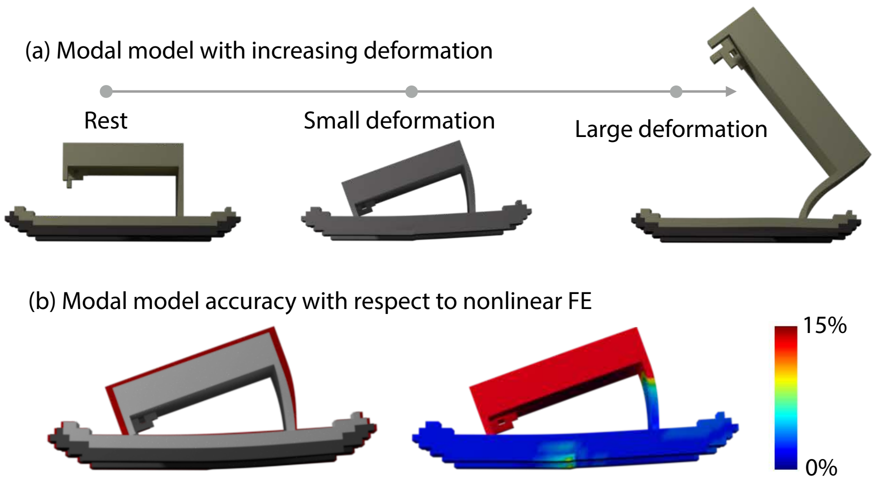

(a) Modal model with increasing deformation
Rest
Small deformation
Large deformation
(b) Modal model accuracy with respect to nonlinear FE
15%
0%

Figure 4-2: Linear modal models suffer from distortion as deformations grow. In (a) we illustrate increasing deformations, left to right, with corresponding inflation errors. Even for smaller deformations, (a) middle, the linear modal model still introduces significant distortions leading to modeling inaccuracies. In (b), left, we overlay simulations of small deformation performed respectively with the linear modal model (red) and the coarsened nonlinear FE model (grey); and, on the right, evaluate accuracy of the modal model. During simulation, even these smaller deformations introduce inaccuracies in the modal model due to element inflation; here up to 13.7%.

objects that undergo rapidly changing boundary conditions and globally varying stress fields due to contacts, loading and impacts. Adaptive meshing in our setting must then, necessarily, feature high numbers of elements to capture these details. See Figure 4-1. Here, using our experimentally validated, accurate element edge length as adaptive meshing threshold, our tetrahedral mesh contains 6.5X to 28X more nodal DoFs than our corresponding DAC-coarsened mesh. The DAC-coarsened mesh is likewise simpler to implement and has no per time step computational cost.

Reduced models utilizing linear modes Hauser et al. [2003], James and Pai [2002] are widely applied to accelerate dynamic simulations. The key issue here is that linear modal models provide only a *linear* approximation of the deformation space leading to inaccurate linearization artifacts, such as swelling during rotation; see Figure 4-2. Optimized quadrature approaches, in turn, can afford efficient integration of non-linear forcing functions, but do not alleviate these artifacts An et al. [2008] when relying on an underlying modal deformation space. Finally, nonlinear modal models Barbič and James [2005] can alleviate some of these issues but so-far remain challenging to incorporate in the design process in comparison to their linear counterparts [Chinesta et al., 2013].

We begin by observing that *numerical coarsening* offers an exciting alternative for efficient yet predictive FE modeling. Coarsening methods effectively apply coarse resolution FE meshes as reduced DoF models and then seek material models that reproduce the behavior of a high-resolution FE counterpart. Analytical solutions for coarsening have been developed for linear material models (models where the stress varies linearly with strain) Kharevych et al. [2009], Nesme et al. [2009], Torres et al. [2016]. Due to this linear assumption, and

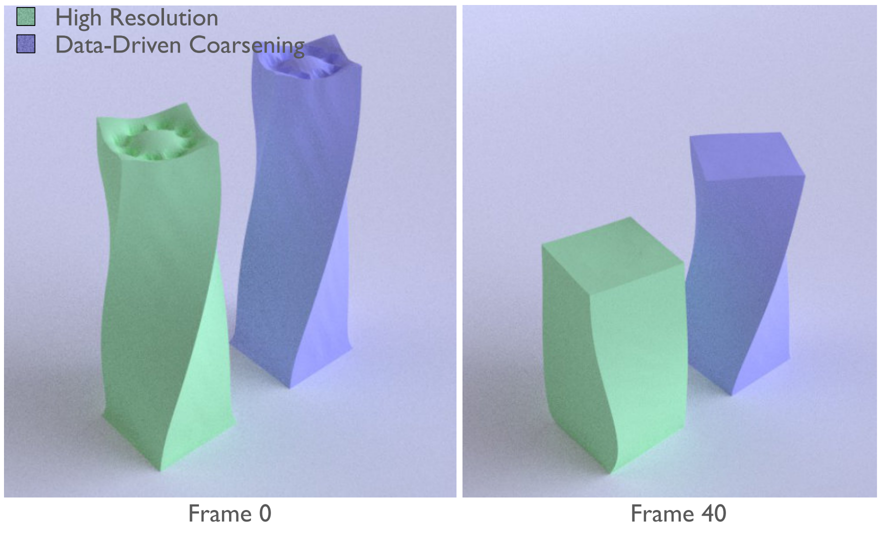


Figure 4-3: Dynamics-oblivious coarsening is significantly inaccurate for dynamic simulation. Left: two twisted elastic bars, initialized to the same configuration, are time-stepped in a dynamic simulation with an energy based, data-driven coarsening (DDFEM) model (blue) and a high-resolution FEM mesh (green). Right: small localized errors in the material model of the DDFEM simulation aggregate across the mesh over time to quickly produce large global errors when compared to the high resolution solution.

similarly to the linear modal models discussed above, we find them difficult to apply for the accurate modeling of the nonlinear materials required for 3D-printed objects. Unfortunately, while coarsening offers promise, prior work, to our knowledge, does not account dynamic effects, inertial properties, nor material damping characteristics. As we see in Figure 4-3, this causes even the nonlinear DDFEM to produce highly inaccurate dynamic simulations. Building on the promise of coarsening techniques and inspired by recent developments in frequency matching for plausible computer animation Li et al. [2014], Wang et al. [2015], we develop a new, dynamics-aware coarsening (DAC) method that, in contrast to prior approaches, provides well-over an order-of-magnitude performance enhancement, while maintaining fabrication-level accuracy when modeling highly dynamic motions subject to frictional contact. Our method does so without complex substructuring, does not require adaptive remeshing, accounts for dynamic effects including damping, and does not introduce prohibitive linear modeling artifacts and so is applicable to a wide selection of nonlinear constitutive models for 3D-printed materials.

4.1.2 Impact Response for Elastic Materials

Even with a suitably accurate FE solution to model *material* dynamics, accurate impact response for elastic materials on collision remains highly challenging. To capture the bounces and rebounds of elastic mechanisms coming into contact with the ground - consider for example the heel strike of a sneaker - we need to get this right. State-of-the-art, implicit time-stepping methods for FEM with contact solve variational forms of time-steppers, e.g., variational Implicit Newmark, subject to additional, fully implicit contact and friction forces Kane

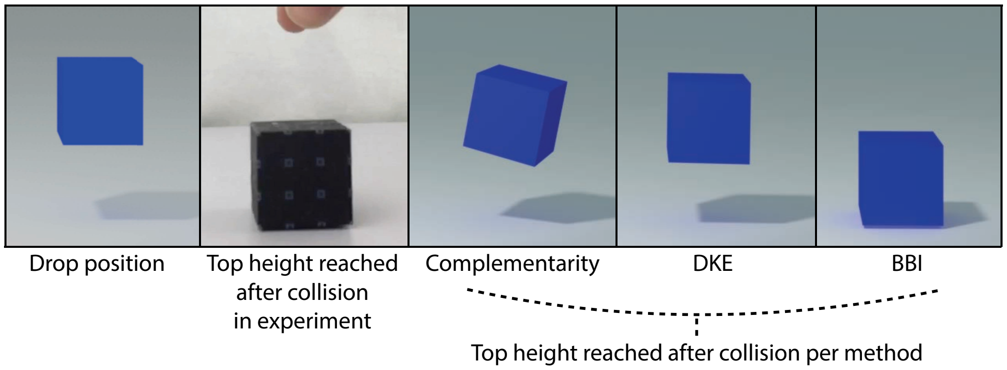

Drop position
Top height reached after collision in experiment
Complementarity
DKE
BBI
Top height reached after collision per method

Figure 4-4: Comparison of complementarity Newmark integration, DKE Stabilization and our BBI model for the impact resolution of an elastic, 3D-printed block. BBI closely matches the maximum rebound height achieved by the experimental result while both the Complementarity and DKE methods overestimate the rebound significantly.

et al. [1999], Pandolfi et al. [2002]. These form so-called nonsmooth or *complementarity* integrators.

However, these complementarity integrators have two well-known flaws Deuflhard et al. [2008]: (1) these methods can yield spurious oscillations on the contact boundaries and (2) the effective impact response of these methods is too energetic. Effectively the normal velocity on impact along the elastic boundary should be dissipated completely. Instead, complementarity methods with Newmark will generate an entirely incorrect elastic restitution; see Figure 4-4. To address these problems Deuflhard et al. [2008] introduced the now-standard DKE contact-stabilization step to filter contact response with projection. In the limit, DKE makes the impact-response model consistent, while in FE codes it is applied as an effective strategy to recover from the well-known limitations of implicit integration with impact. However, it remains widely acknowledged that the right-way to accurately model high-speed collision response with implicit FE remains an open question at this time - not only in graphics - but more broadly in scientific computing as well. As an example consider Figure 4-4 where we see that both the complementarity model *and* the DKE filter produce different but equally incorrect predictions of the response of a stiff elastic block dropped on the floor. Here we offer a new, Boundary-Balancing Impact (BBI) model for FE that gains us accurate prediction of impact response for the stiff 3D-printed materials we focus on here; see Figure 4-4.

4.1.3 Summary and Contributions

High-fidelity simulation methods for elastodynamics are too slow for use in fabrication design tasks while existing strategies to reduce the simulation cost of elastica (including adaptive, reduced and coarsened models) are too inaccurate and/or too expensive to employ. Finally, existing FE models for simulating elastic collision and rebound miss critical compliance coupling in the filter stage.

We have exposed and analyzed the limitations of simulation methods for the predictive modeling of elastodynamics at rates sufficient for fabrication design optimization. Next, we develop our *Dynamics-Aware Coarsening* (DAC) method to address this need (Section 4.3).

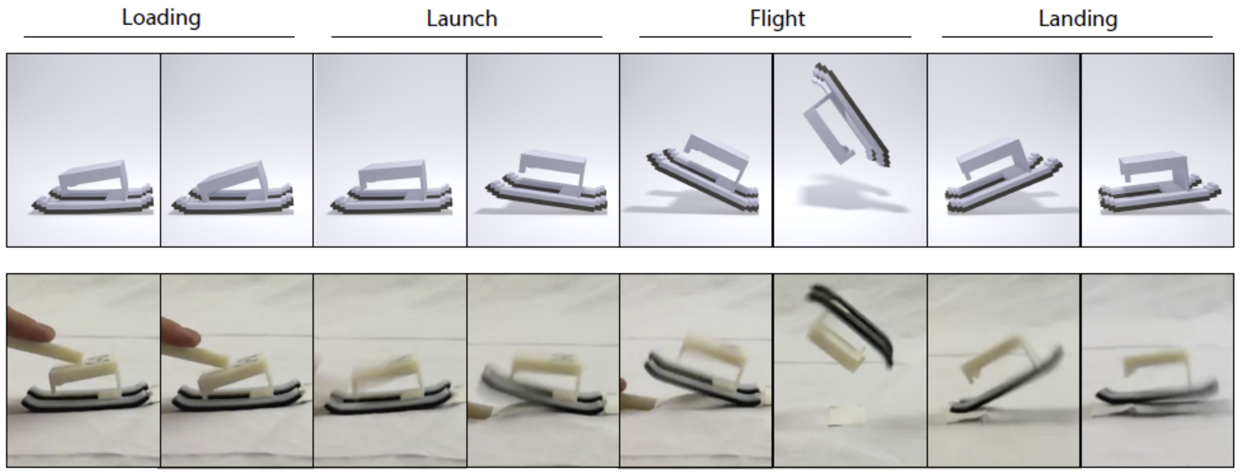


Figure 4-5: Bottom: the dynamic behavior of a 3D-printed, elastic jumping mechanism in experiment. Top: our DAC-coarsening combined with our BBI impact model generate a simulation that predictively captures this experimental behavior at rates 79X faster than state-of-the-art FEM.

DAC jointly identifies *and* predictively simulates fabricated materials. To address current limitations in FE collision-response filtering we then introduce our *Boundary Balancing Impact* (BBI) model (Section 4.4). We then validate these contributions by comparing simulated results generated by DAC and BBI with real-world results experimentally obtained from a range of compliant 3D-printed jumping and throwing mechanisms that flip, throw projectiles, jump onto obstacles and jump over walls (Section 4.5.4).

4.2 Simulation Preliminaries

Accurate time-varying tracking of energy dissipation is key to accurate dynamic simulation Marsden and West [2001] and so we rely on implicit Newmark time integration, a discrete variational integrator Kane et al. [2000] with consistent high-quality energy tracking. We experimented with other integrators, including linearly implicit Newmark, Implicit Euler, and BDF2, but found them to be wanting in either stability or energetic behavior; see our Supplemental for details.

4.2.1 Discrete Model

We begin with a standard elastic material model for 3D printed materials. To capture stiff elastic response of 3D-printed materials we use the neo-Hookean material model, augmented with Rayleigh damping to capture transient dissipation of vibrations, and discretize with eight-node hexahedral finite elements. The continuous equations of motion derived from Newton's second law is

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{F}(\mathbf{q}) + \mathbf{D}(\mathbf{q})\mathbf{v}, \tag{4.1}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{F}(\cdot)$ is the internal force vector, $\mathbf{D}(\cdot) = a\mathbf{M} + b\mathbf{K}(\cdot)$, with $a, b \in \mathbb{R}^+$ is the Rayleigh damping matrix, $\mathbf{M}$ is the stiffness-consistent mass-matrix [Belytschko et al., 2013], $\mathbf{K}(\cdot) = -\nabla\mathbf{F}(\cdot)$ is the tangent stiffness matrix, and $\mathbf{q}, \mathbf{v} \in \mathbb{R}^{3n}$ are respectively the nodal position and velocity vectors. We start with a second-order, implicit Newmark time discretization

$$\begin{aligned} \mathbf{v}^{t+1} &= \mathbf{v}^t + \frac{h}{2}(\dot{\mathbf{v}}^t + \dot{\mathbf{v}}^{t+1}), \\ \mathbf{q}^{t+1} &= \mathbf{q}^t + \frac{h}{2}(\mathbf{v}^t + \mathbf{v}^{t+1}), \end{aligned} \tag{4.2}$$

where h is the timestep size. For convenience, we define

$$\boldsymbol{\delta}^{t+1} = \mathbf{q}^{t+1} - \mathbf{q}^t. \tag{4.3}$$

Substituting Equation 4.2 and 4.3 into Equation 4.1, we arrive at the following discrete dynamics equation to solve for the unknowns $\mathbf{q}^{t+1}$ and $\mathbf{v}^{t+1}$

$$\mathbf{M}\boldsymbol{\delta}^{t+1} = \mathbf{b}^t + \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^{t+1}) - \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^{t+1})\mathbf{v}^{t+1}, \tag{4.4}$$

$$\begin{aligned} \mathbf{q}^{t+1} &= \mathbf{q}^t + \boldsymbol{\delta}^{t+1}, \\ \mathbf{v}^{t+1} &= \tfrac{2}{h}\boldsymbol{\delta}^{t+1} - \mathbf{v}^t, \end{aligned} \tag{4.5}$$

where

$$\mathbf{b}^t = h\mathbf{M}\mathbf{v}^t + \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^t) - \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^t)\mathbf{v}^t. \tag{4.6}$$

In the absence of dissipative forces this method is symplectic and momentum preserving [Kane et al., 2000]. With dissipation we find that integration gives us accurate bookkeeping of system energy at comparable cost to implicit Euler.

4.2.2 Material Parameters

No matter how good our energy bookkeeping, the overall fidelity of our method is critically determined by the accuracy of the material parameters we select. While many material parameters are reported in the literature, there remains large and significant variation in these values across 3D-print batches, printing orientations and curings; see §5. For predictive simulation we need to identify these values. In addition, we must model dissipation requiring us to determine unreported damping properties; e.g., a and b in the Rayleigh model. Finally, as discussed below and complicating matters even further, these material parameters are discretization dependent at non-convergent spatial resolutions. In the next section we will detail our DAC model to capture dynamic deformation, stiffness and damping at coarsened spatial resolutions.

4.2.3 Contact and Friction

For contact we need to model nonpenetration constraints and frictional contact forces that resist sliding along interfaces. Contacts are between object parts or between a part and a fixed boundary such as the ground. At each time step we apply continuous collision detection to the predicted trajectory to gather contact constraints into a contact set $\mathcal{C}$.

To simplify the following, for each such contact $k \in \mathcal{C}$, the relative acceleration between material points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ (at contact k) can be expressed via the map $\boldsymbol{\Gamma}_k : \dot{\mathbf{q}} \to \dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_j$. See our Supplement for details on construction of $\boldsymbol{\Gamma}_k$. If $\boldsymbol{y} \in \mathbb{R}^3$ is a force applied to point $\boldsymbol{x}_i$, and an equal but opposite force is applied to point $\boldsymbol{x}_j$, then $\boldsymbol{\Gamma}_k^T \boldsymbol{y}$ is the resulting generalized force applied to the contacting system.

In turn, points in contact apply an equal and opposite force along their shared, unit-length normal $\boldsymbol{n}_k \in \mathbb{R}^3$. In global coordinates this is equivalent to applying a force of magnitude $\bar{\alpha}_k \in \mathbb{R}^+$ along a generalized normal

$$\mathbf{n}_k = \boldsymbol{\Gamma}_k^T \boldsymbol{n}_k \in \mathbb{R}^{3n}, \tag{4.7}$$

to the system. The subspace of generalized normal directions

$$\mathbf{N} = (\mathbf{n}_1 ... \mathbf{n}_{|\mathcal{C}|}) \tag{4.8}$$

then forms a basis for contact forces. Concatenating the corresponding force magnitudes in $\alpha = (\bar{\alpha}_1, ..., \bar{\alpha}_{|\mathcal{C}|})^T$, the total contact force applied in the system is then $\mathbf{N}\alpha$.

Friction forces lie in the tangent plane orthogonal to the contact normal. At each contact k we sample an orthogonal pair of unit length vectors from the tangent plane. The 3×2 matrix composed column-wise of these samples is given by $\boldsymbol{T}_k$ so that a friction force, $\boldsymbol{f}_k \in \mathbb{R}^3$, applied at a contact k, lies in the span of $\boldsymbol{T}_k$ with $\boldsymbol{f}_k = \boldsymbol{T}_k \bar{\beta}_k$, where each $\bar{\beta}_k \in \mathbb{R}^2$ gives the frictional response coefficients at contact k.

The total friction force applied to the system at each contact k must be equal and opposite and is $\mathbf{f}_k = \boldsymbol{\Gamma}_k^T \boldsymbol{T}_k \bar{\beta}_k$. The generalized basis for a friction force at contact k is then

$$\mathbf{T}_k = \boldsymbol{\Gamma}_k^T \boldsymbol{T}_k \in \mathbb{R}^{3n \times 2}. \tag{4.9}$$

We build the corresponding subspace of generalized tangent directions,

$$\mathbf{T} = (\mathbf{T}_1 ... \mathbf{T}_{|\mathcal{C}|}) \tag{4.10}$$

and form the corresponding vector of frictional force coefficients as $\beta = (\bar{\beta}_1^T, ..., \bar{\beta}_{|\mathcal{C}|}^T)^T$. The total friction force on the system is then $\mathbf{T}\beta$.

Contact and friction forces can be inexpensively modeled explicitly [Belytschko et al., 2013] but this introduces instabilities and nonphysical oscillations on boundaries even at small time step for stiffer materials Deuflhard et al. [2008]. Thus FEM state-of-the-art generally turns to implicit time-integration for efficient contact force modeling.

Kane et al. [1999] proposed the now standard nonsmooth-Newmark method for contact modeling. This is a fully implicit time-stepping model that couples frictional contact with internal energies and forcing in each solve.

$$\mathbf{M}\boldsymbol{\delta}^{t+1} = \mathbf{b}^t + \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^{t+1}) - \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^{t+1})\mathbf{v}^{t+1} + \tfrac{h^2}{2}\mathbf{N}\alpha + \tfrac{h^2}{2}\mathbf{T}\beta. \tag{4.11}$$

Note that here, unlike internal forces, contact forces are evaluated solely at the time step endpoint to ensure dissipation. Next, to fully define the time stepper, consistency conditions are required for contact and friction forces.

Enforcing the complementarity model for contact requires contact forces to balance along boundaries

$$0 \leq \alpha \perp \mathbf{N}^T \delta^{t+1} \geq 0, \tag{4.12}$$

Friction in turn is modeled with the *Maximal Dissipation Principle*, requiring friction to maximize the rate of negative work done at each contact, $-\boldsymbol{f}_k^T \dot{\boldsymbol{x}}$. The total dissipation performed by friction is then

$$\sum_{k \in \mathcal{C}} \big(-\boldsymbol{f}_k^T \dot{\boldsymbol{x}}_k\big) = -\Big[\sum_{k \in \mathcal{C}} \boldsymbol{f}_k^T \boldsymbol{\Gamma}_k\Big] \dot{\mathbf{q}} = -\beta^T \mathbf{T}^T \dot{\mathbf{q}}. \tag{4.13}$$

Then, maximizing the Coulomb-constrained dissipation simultaneously at *all* contact points, with the implicit Newmark discretization Pandolfi et al. [2002], gives us the final condition for our numerical integration

$$\min_{\beta} \{\beta^T \mathbf{T}^T (\tfrac{2}{h} \delta^{t+1} - \mathbf{v}^t) : \mu_k \bar{\alpha}_k \geq \|\bar{\beta}_k\|, \, \forall k \in \mathcal{C}\}, \tag{4.14}$$

where μ_k is the local friction coefficient at contact k.

Taken together stepping this implicit *complementarity* Newmark integrator with (4.6), (4.11), (4.12), (4.14) and (4.5) provides predictive simulation at slower speeds of contact. For higher speed contacts and impacts, the remaining challenge lies in stabilizing contact stresses, velocities and displacements Deuflhard et al. [2008] to ensure that simulated objects' bounces and rebounds consistently match with their real-world counterparts. In Section 4.4 we will address this issue with a new impact model for FE simulation. First, however, we address our fundamental scaling problem: how can we gain accurate dynamic simulation without being bottlenecked by systems too large to solve quickly per time step?

4.3 Dynamics-Aware Coarsening

We couple numerical coarsening with parameter acquisition. Our DAC method computes *numerical* stiffness parameters for the nonlinearly elastic, Neohookean material model, and damping parameters for the Rayleigh damping model so that, when applied to the coarsened simulation mesh, the dynamic behavior of a high-resolution simulation is preserved.

Because the goal of DAC is to replicate the dynamic behavior of a high-resolution simulation, we focus on creating a coarse mesh which captures both the large-scale deformation modes, and the corresponding natural frequencies of the high-resolution mesh. We do this in two stages. First, we produce a coarse hexahedral mesh that can replicate the large scale deformation modes of our high-resolution mesh. Second, we compute material and damping parameters that yield matching fundamental frequencies for each mode shape on this coarse mesh.

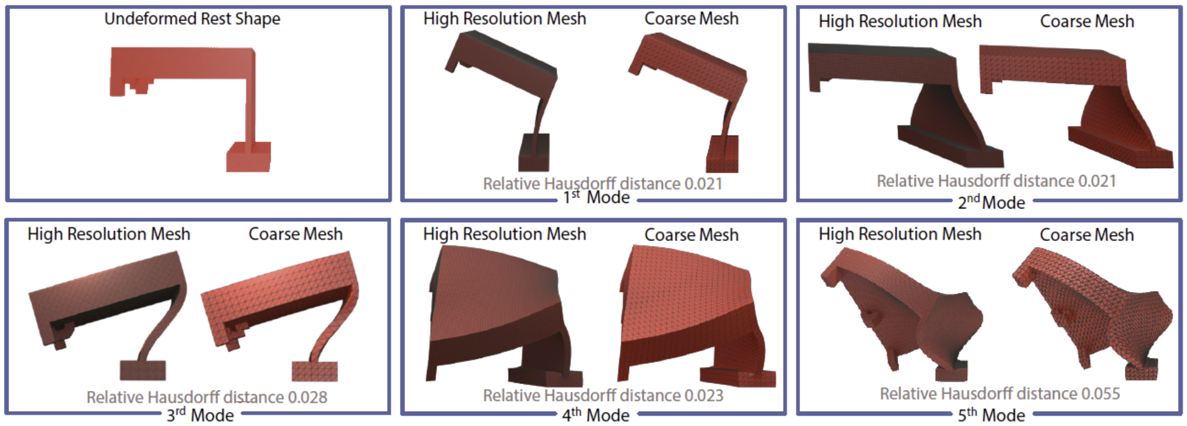


Figure 4-6: Dynamics-aware coarsening (DAC) coarsens meshes to capture deformation with calibrated stiffness. We observe FE meshes capture accurate deformation modes up to a quite coarse resolution. However, these same coarse meshes suffer from numerical stiffening (see Figure 4-7). We design coarse mesh FE models by matching both significant deformation modes (above) and their captured material response (see Figures 4-10 and 4-20) to obtain efficient and predictive coarse-mesh FEM simulations of dynamics.

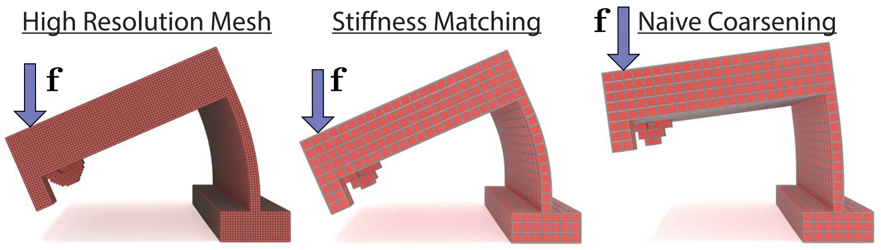


Figure 4-7: Static Deformation Test. We apply an identical load to three meshes with the same model geometry. With the same material parameters, a high-resolution mesh (left) is effectively 2.5x softer than the corresponding coarse mesh (right). Applying our captured numerical Young's modulus to the coarse mesh (middle) regains the correct deformation of the original, high-resolution mesh on the left.

4.3.1 Geometric Coarsening

DAC uses an iterative procedure to create the coarse mesh while maintaining mode shapes. We initialize our mesh to a coarse hexahedral discretization of the starting geometry, $\mathbf{q}^0$, and then subdivide recursively until we reach a convergent mesh resolution. We then solve the generalized mass-PCA system

$$\mathbf{K}(\mathbf{q}^0)\mathbf{q} = \lambda \mathbf{M}\mathbf{q} \tag{4.15}$$

for the dominant shape modes of the convergent system and then coarsen via bisection with mass-PCA until we reach a maximally coarse mesh that matches the dominant four shape modes to tolerance. We use a relative geometric difference of 5% (Hausdorff distance) as our tolerance threshold. Typically this is a short validation step as even the coarsest meshes generally satisfy this criteria (Figure 4-6).

4.3.2 Material Parameter Fitting

Our geometric coarsening ensures that our DAC mesh captures significant deformation modes of our design accurately. However, when simulated, these same coarse meshes suffer from *numerical stiffening* – an increase in effective stiffness and damping as a consequence of decreased mesh resolution; see Figure 4-7. This leads to unacceptably inaccurate simulated trajectories no matter how we simulate this system. Regaining predictive stiffness and damping by refining the discretization would take us back to intractable mesh sizes; see Figure 4-1. Rather than refine, we keep our coarse mesh (as we have already ensured that our geometry is resolved there) and instead calibrate its frequency spectrum to directly match experiment.

To do this, we rescale our coarse model stiffness so that its fundamental frequency matches an observed frequency. As we will see below, this simple analysis sufficiently recovers effective stiffness and damping to regain a predictive nonlinear simulation with our coarse mesh.

From the geometric coarsening step above, we retain the numerical eigenvalues, λ_i^0, of our coarse mesh with corresponding deformation modes m_i approximated linearly by the damped harmonic oscillator Shabana [2012]

$$\ddot{m}_i = -(a + b\lambda_i)\dot{m}_i - \lambda_i m_i, \tag{4.16}$$

or equivalently

$$\begin{aligned} m_i(t) =& A_i \exp(-d_i t)\sin(2\pi f_i t + \theta_i), \\ f_i =& \frac{1}{2\pi}\sqrt{\lambda_i - (\frac{a + b\lambda_i}{2})^2}, \\ d_i =& \frac{1}{2}(a + b\lambda_i), \end{aligned} \tag{4.17}$$

where a and b are Rayleigh Damping parameters and λ_i is the i^{th} eigenvalue associated with the i^{th} deformation mode.

We then 3D-print calibration rigs with tracker markings, see e.g., Figure 4-8, and capture high-speed video (240 fps) of the rigs vibrating. We extract a tracked trajectory m_t of the

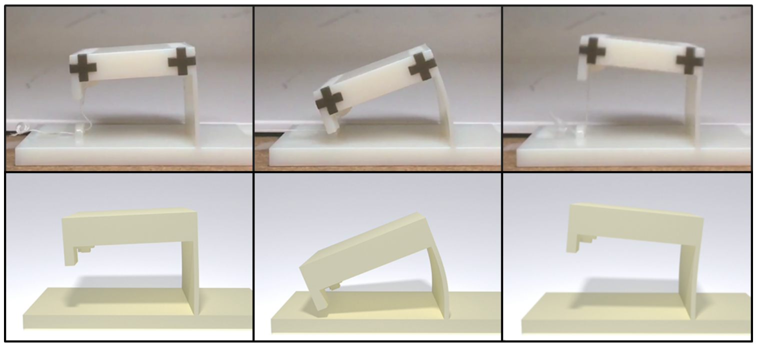


Figure 4-8: Top: tracking the oscillations of a 3D-printed calibration rig allows us to measure mesh-dependent stiffness and damping parameters. Bottom: the resulting DAC-simulated frames at corresponding times for visual comparison to the captured motion.

marker motion from the video capture and solve the inverse harmonics problem [Mandelshtam and Taylor, 1997] to find the printed beam's frequency, f_t, and damping, d_t, parameters. Setting the tracked f_t and d_t in (4.17) and $a = 0$, based on our observation of minimal effective mass-damping, simultaneously retrieves the captured target eigenvalue λ_t and the unknown and unreported stiffness damping parameter b required for dynamic simulation.

With our tracked λ_t in hand, our final step is to map the initial material Young's modulus, E_m, that we use to compute λ_i^0 (we set $E_m = 1$ throughout) to a new, *numerical* modulus value, E_n. We seek an E_n that will match the numerical stiffness response of the simulated coarse FE mesh to the captured material response. We do this with a simple argument of fixed proportionality between the principle eigenvalues and the moduli by setting

$$E_n \leftarrow \frac{\lambda_t}{\lambda_1^0} E_m. \tag{4.18}$$

As validation we confirm that our coarsened FE simulations, initialized to the starting calibration pose, with Young's modulus set to E_n, and stiffness damping set to the measured damping parameter, match both high-resolution simulation and the tracked calibration rig, up to viscosity—which we so far find unnecessary to model; see Figure 4-20.

Note that, in our experiments, increasing the number of mode shapes which must fall below our error threshold of 5% Hausdorff distance does not improve simulation accuracy. Figure 4-10 shows a comparison of DAC meshes created using 4 and 10 modes for the error threshold. The resulting simulations are indistinguishable from each other and both match measured experimental data equally well.

In practice we observe that DAC captures object motions containing large contributions from a number of modes. To understand why we see that DAC scaling corrects frequencies of modes well beyond the first, so that, for example, for our plant and walker models, DAC reduces average frequency error across the first 10 modes from 27.1% to 2.4%.

Figure 4-9 shows an example of DAC coarsening applied to plant and flamingo models. With DAC we accelerate the simulation of the time varying deformation of our plant object by 25X while achieving good approximation to the high-resolution simulation mesh; see Figure 4-11. Here Figure 4-11 distinguishes between convergent FEM discretizations and

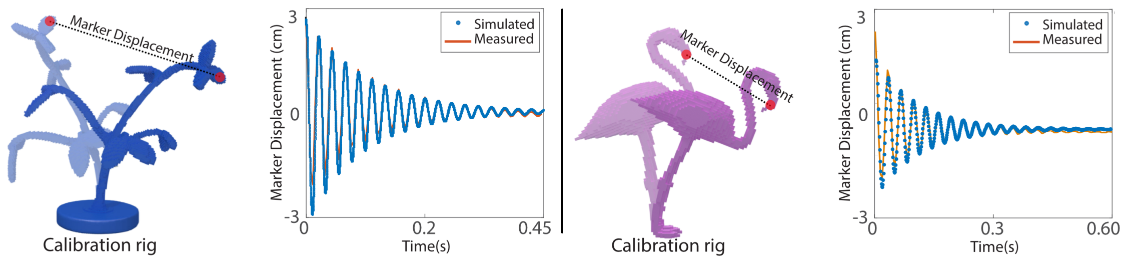


Figure 4-9: Measured and simulated vibrations of the 3D-printed plant (digital material) and 3D-printed flamingo (Rigur RGD450) models (magnify the plots for details of fits).

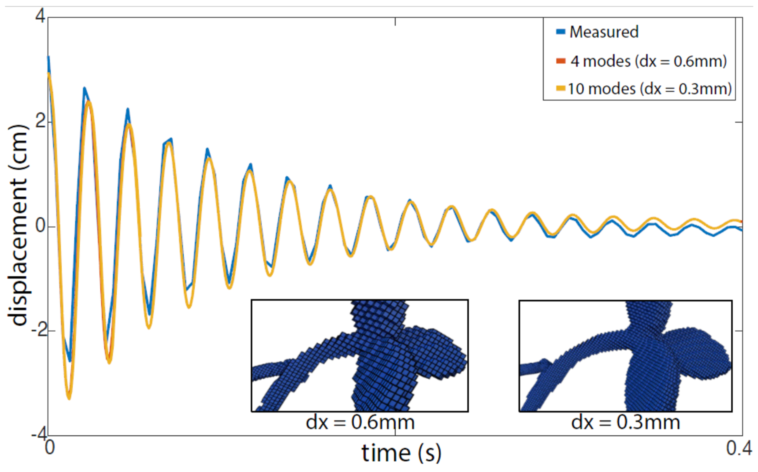
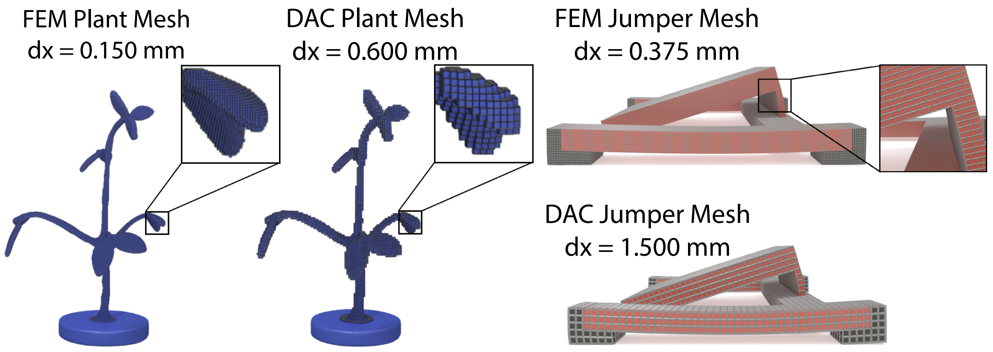


Figure 4-10: DAC coarsening comparison for increasing the number of fitted modes. Our default setting requires less than 5% relative Hausdorff distance up to the 4^{th} mode and gives a coarsened FE mesh with $dx = 0.6mm$. If we increase the number of modes for our DAC fit to less than 5% relative Hausdorff for up to the 10^{th} mode shape, we instead obtain a mesh with $dx = 0.3mm$. Note that the resultant simulations for these two DAC meshes are indistinguishable (magnify the plot for detail of fit).



Simulation	Model	dx(mm)	Timing avg/step(s)	Memory (G)	Elements
FEM	Plant	0.100	1876.3	48.5	1,004.6K
FEM	Plant	0.150	226.8	11.9	297.9K
DAC	Plant	0.600	9.2	0.4	16.1K
FEM	Jumper	0.150	—	379.7	5,450.0K
FEM	Jumper	0.375	791.0	18.0	346.3K
DAC	Jumper	1.500	10.4	0.3	5.4K

Figure 4-11: Statistics for our DAC simulations compared with two choices of accurate FE: a convergent FE model, and a validated accurate FE model (validated as matching experimental behavior and modal frequencies are within 5% of convergent values). For each simulation we report the model used (plant and jumper models), the mesh element size, the average wall-clock time spent per dynamic time step, memory usage, and the number of elements in the simulated mesh. Timings were recorded on an Intel Xeon E5-2666 v3, 2.9Ghz with 4 CPU threads.

accurate FEM discretizations for meshes used in our examples. Convergent discretizations are ones for which the spatial resolution is high enough so that the modal frequencies of the mesh have converged to their final values (changing by less than 1% with respect to previous subdivision). Accurate discretizations are ones for which the resolution of the FE mesh is such that the modal frequencies are within 5% of convergent values. In this paper all results are with respect to the coarser, accurate FE meshes. Even compared to these we achieve speedups of up to 79X. We also note that, if measurement data is not available, DAC calibration can be carried out using high-resolution simulation data.

4.4 Boundary Balancing Impact Model

With our DAC discretization in place, we will now derive a new, Boundary-Balancing Impact (BBI) model for FE that gains us accurate prediction of impact-response for the 3D-printed elastic materials we focus on in this work.

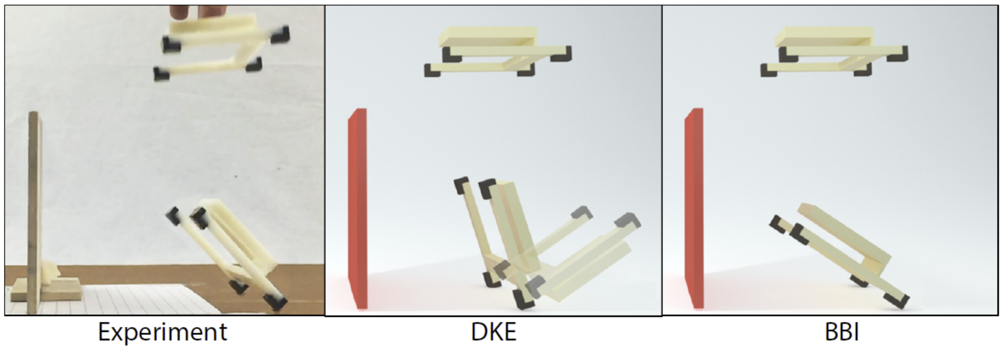


Figure 4-12: Impact model testing with a 3D-printed jumper dropped onto the ground. We show overlaid frames captured, left to right, from an experiment (left) and two corresponding simulations (at $h = 10^{-4}s$) that respectively apply DKE (middle) and BBI (right) impact models. In experiment the fabricated jumper lands, bounces up and then rests upright on its feet (left). However, simulation with the DKE model (middle) rebounds too high, flips over and so incorrectly predicts that the jumper will fail by landing on its back after collision (middle). With our BBI model (right), our simulation qualitatively predicts the experimentally determined landing behavior for this design.

4.4.1 Complementarity Integration Revisited

The nonsmooth Newmark complementarity integrator we reviewed in Section 4.2 has several well-known flaws Deuflhard et al. [2008] that we illustrate next. In Figure 4-13 right, we drop a 3D-printed block from height of 3 cm onto a flat surface. As it is both stiff and highly damped it lands without perceptible rebound. Yet, when we simulate the same drop of the block with the Newmark complementarity integrator the block rebounds up to a height of 0.89 cm; see Figure 4-13 left. Errors on this scale are unacceptable in a fabrication design process where they can make the difference between success and failure - see Figure 4-12.

What is going wrong in these examples? The Newmark discrete velocity update step in (4.5) gives rise to an arbitrary, undesirable (and for elastic materials) generally incorrect choice of restitution. Consider the impact of our material at contact with a normal $\mathbf{n}$. Here the complementarity constraints ensure that the new displacement δ^{t+1} along this normal are zero so that $\mathbf{n}^T\delta^{t+1} = 0$. However, although this nicely satisfies position constraints, upon substitution we see that $\mathbf{n}^T\mathbf{v}^{t+1} = -\mathbf{n}^T\mathbf{v}^t$ so that Newmark gives an incorrect, fully elastic (coefficient of restitution = 1) effective impact response. Yet the impact response for elastica along an impact boundary should instead be inelastic with the normal velocity on impact along the elastic boundary dissipated completely Doyen et al. [2011]. In turn this results in much too large rebounds upon impact as we observe in Figures 4-12 and 4-13. A related error for complementarity integrators manifests in commonly observed spurious oscillations in positions and tractions along contact boundaries. These oscillations are the combined result of instabilities in contact stresses, velocities and displacements. Notably, both of these issues arise with arbitrary impact geometries, not just in the planar example discussed here.

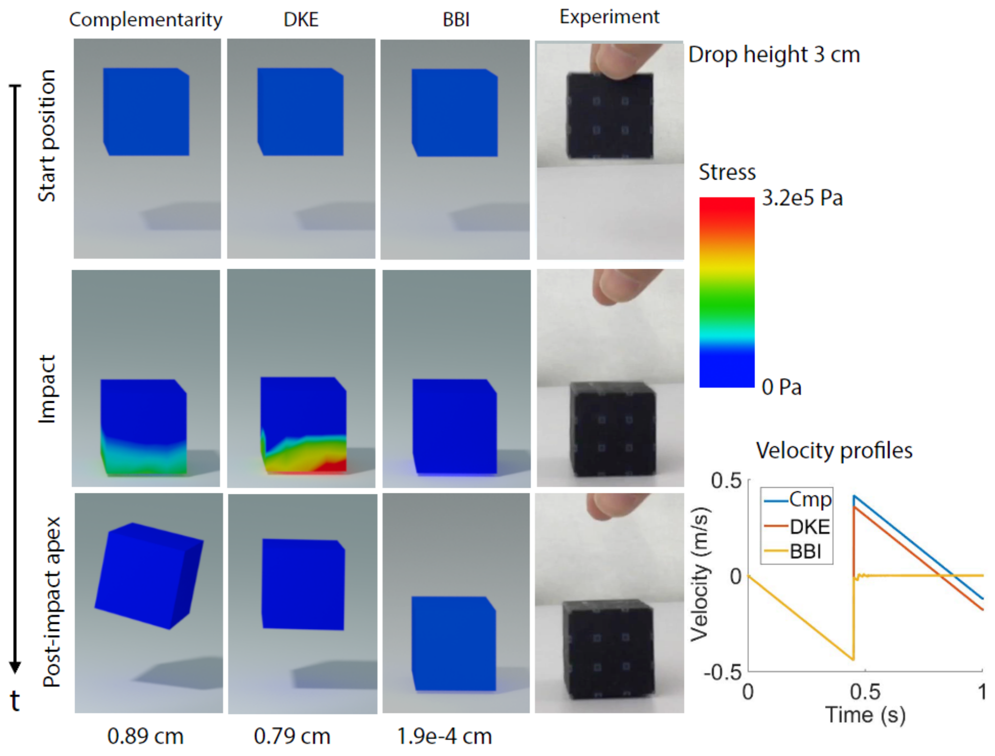


Figure 4-13: Impact validation test. A 3D-printed, stiff elastic block is dropped face-first on the ground. Left-to-right we compare the simulated results of three FE impact models - complementarity, DKE and our BBI model - with experimental results. We show configuration, at start, impact, and post impact maximum rebound height with details on stress distribution at impact, apex height reached, and (inset) the velocity profile for each simulation. As the effective restitution of elastica varies with angle of impact, see below in Figure 4-14 for a comparable oblique drop experiment.

4.4.2 DKE Contact Stabilization

To address these widely reported problems, Deuflhard et al. [2008] introduced a contact stabilization step to filter contact response with projection. They observe that contact forces acting on the material boundary should be balanced and so proposed a now-standard FEM contact-stabilization filter (DKE) that applies an L2-projection that zeroes out normal displacements along the boundary of materials at contact interfaces.

This projection on displacement is performed at the end of each time step, after the position update (4.11) has been solved. This ensures a correct *inelastic* response at the contact boundary and is effective in producing desirably stabilized contact tractions Deuflhard et al. [2008], Krause and Walloth [2012]. Nevertheless, when we compare DKE against experiment (Figures 4-12, 4-13, and 4-15), we see that the DKE projection likewise introduces unacceptably large rebound errors when compared with real-world results.

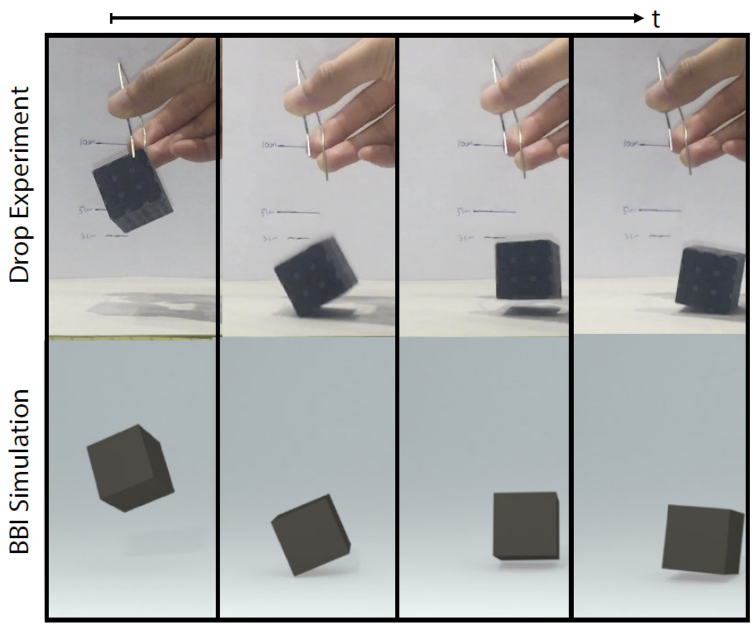


Figure 4-14: Comparison of our BBI model to experiment for a 3D-printed cube dropped from an oblique initial orientation.

4.4.3 Boundary Balancing Impact

To better understand why DKE projection does such a poor job of resolving impacts in our elastic materials, let us consider again our simple dropped block example in Figure 4-13. We observe that by projecting out just the normal component of displacement along the boundary the DKE method artificially *concentrates* high stresses along the elements just inside the boundary. This can be seen in the impact row of Figure 4-13. These concentrated stresses effectively load the near-boundary layers which then spring back, introducing a much too large response as seen in the final row of Figure 4-13. The problem here is that the L2-projection applied is material-oblivious and yet material properties clearly mediate impact response. Compliance distributes contact stresses quickly through an elastic material while, in damped materials, internal friction rapidly attenuates the response.

With these observations in mind we define a new Boundary Balancing Impact (BBI) model that can effectively impose boundary force-balance in a material-aware fashion. Starting with our base time integrator we define a compliant, discretization- and material-aware metric for projection below. The resulting stabilizing impact model better duplicates results in our design applications and experiments. Upon completing each time step from $t-1$ to

t we replace the Newmark update in (4.5) with a compliant boundary projection update

$$\begin{aligned}\mathbf{c} &= 2\boldsymbol{\delta}^{t+1} - h\mathbf{v}^t,\\ \mathbf{A} &= \mathbf{M} + \frac{h^2}{4}\mathbf{K}(\mathbf{q}^{t+1}) + \frac{h}{2}\mathbf{D}(\mathbf{q}^{t+1}),\\ \mathbf{d}^* &= \underset{\mathbf{d}}{\operatorname{argmin}}\left\{\frac{1}{2}\|\mathbf{d}-\mathbf{c}\|^2_{\mathbf{A}} : \mathbf{N}^T\mathbf{d} \geq 0\right\},\\ \mathbf{v}^{t+1} &= \frac{1}{h}\mathbf{d}^*,\\ \mathbf{q}^{t+1} &= \mathbf{q}^t + \boldsymbol{\delta}^{t+1}.\end{aligned} \tag{4.19}$$

This new impact model projects the explicit predicted velocity displacement, $\mathbf{c}$, to the nearest set satisfying force balance on the boundary with respect to the local approximation of both material *stiffness* and *damping.* We find that this effectively distributes the contact-stabilized displacement across the material from the boundary layers. The effect of impact is communicated to the material interior while still ensuring that impacts are correctly inelastic along the active contact boundary. This leads to accurate predictions of real-world bouncing and rebounds. In our simple drop test we see in the impact row of Figure 4-13 that, at impact, response for this damped material has been correctly dissipated and the resulting normal displacement closely matches real world results in the apex row of Figure 4-13.

As we consider a range of impact angles, as well as impact with more complex geometries, multi-material 3D-prints and self contact we see that BBI still consistently and more accurately captures impact-response behavior. See Figures 4-12-4-16. These examples demonstrate the range and complexity of responses obtained by BBI coupling impact to stiff elastic materials with large deformations as well as objects composed of multiple materials undergoing both self-contact and stiction.

4.5 Results and Discussion

To test our proposed algorithms we compare simulated results generated by DAC and BBI with real-world results experimentally obtained from a range of compliant 3D-printed jumping and throwing mechanisms. For each mechanism we begin with a dynamic, time-varying goal, e.g., for our *jump-over* goal: "when pressed and released, jump over a given wall and land upright on the far side" (see Figure 4-24).

Simulations suitable for fabrication design must accurately predict both when a design fails and when it succeeds, thus we require real world examples of each. Below we outline our approach to creating these mechanism examples for testing, discuss our identification results, and review our implementation. We then detail our experiments comparing DAC and BBI's simulated predictions for each design task below in Section 4.5.4 and validate outcomes of the simulations against repeated user trials in a study discussed in Section 4.5.5.

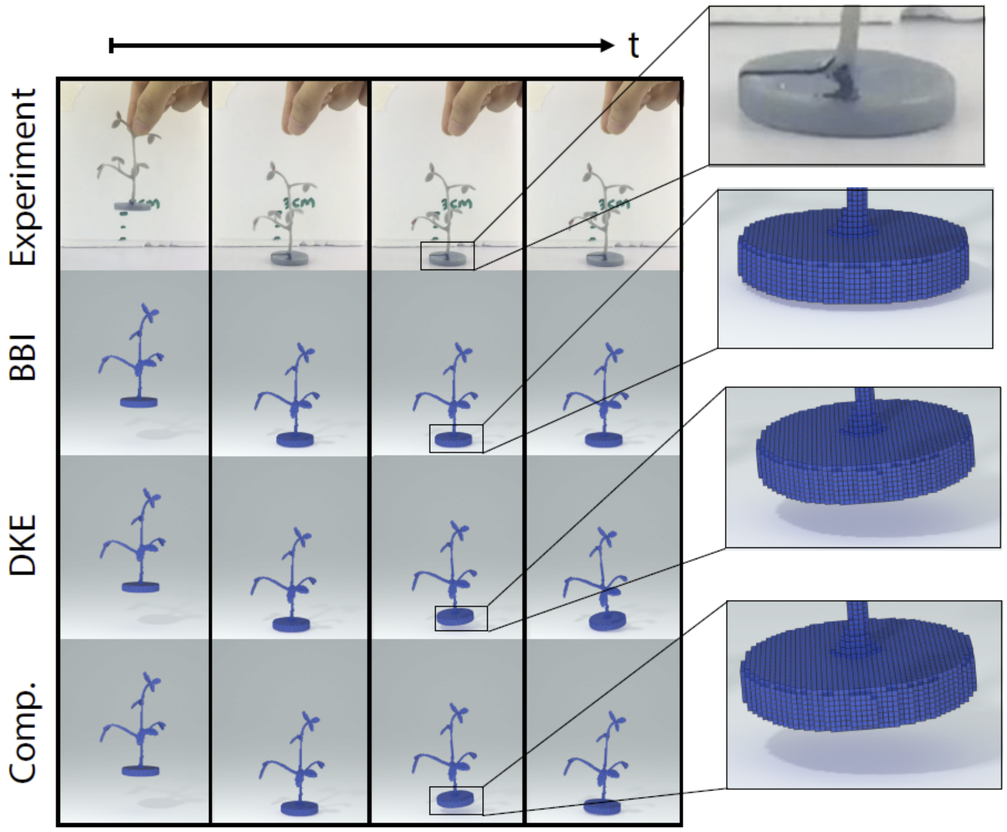


Figure 4-15: A drop test of the 3D-printed plant model (digital material). We compare simulated results from Complementarity, DKE and our BBI model with experiment. BBI solely reproduces the observed impact behavior.

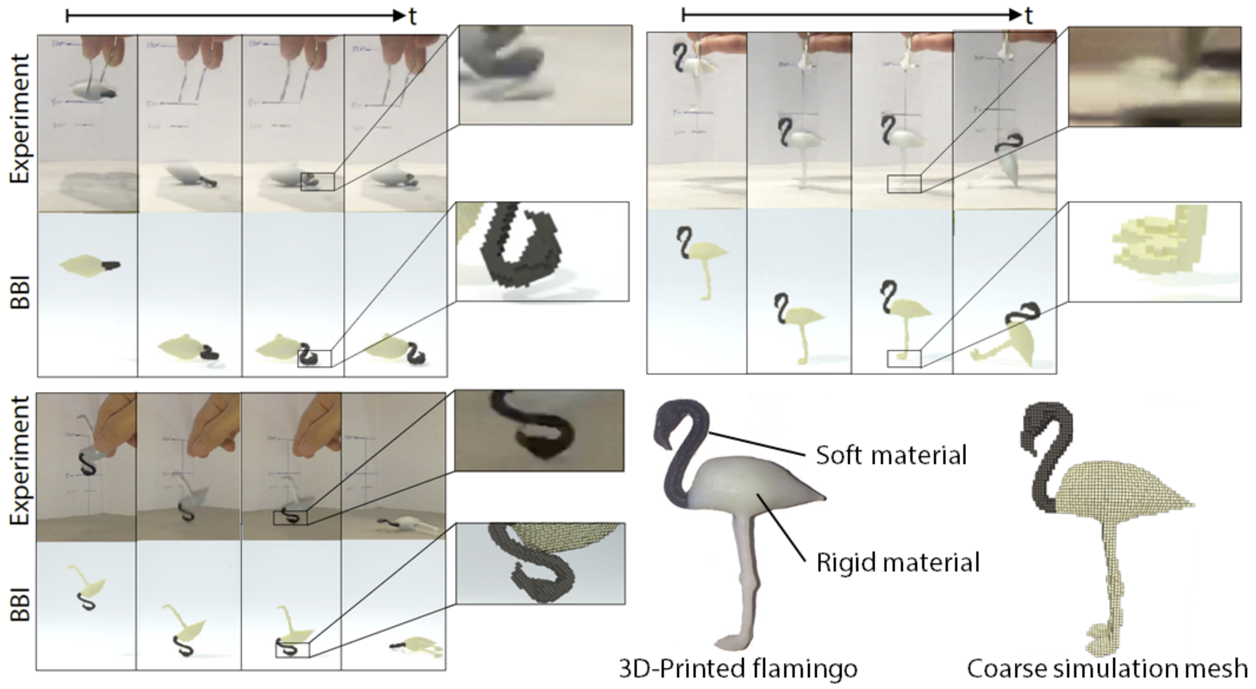


Figure 4-16: Drop tests of a multi-material 3D-printed flamingo model (Rigur and TangoBlackPlus) from a variety of orientations. While effective restitution of elastica varies with angle of impact, in all cases our BBI simulations agree with experiment.

4.5.1 Creating Mechanism Examples

We focus our tests here on jumping-related mechanisms. The analysis and design of jumping mechanisms is an increasingly active [Bergbreiter, 2008, Bergbreiter and Pister, 2007, Bingham et al., 2014, Churaman et al., 2011, Jung et al., 2014, Koh et al., 2013, 2015, Li et al., 2015, Vella, 2015], challenging and practical domain that incorporates high-speed transient dynamics of stiff elastic materials undergoing impact and so is an ideal test case for DAC and BBI. Research in jumping mechanisms has focused on efficient energy transfer into jump height, see e.g., Noh et al. [2012] with aligned research on a range of approaches for controlled jumping [Bartlett et al., 2015, Li et al., 2015, Loepfe et al., 2015]. In all cases, to our knowledge, mechanisms have been developed via costly, manual iterations of hands-on experiment, re-design, fabrication and one-off simulations [Bartlett et al., 2015, Cho et al., 2009], while even the stable landing of dynamic jumps has remained highly challenging [Jung et al., 2015].

For each of our jumping and throwing goals (see Section 4.5.4 below) we first attempted to manually create successful designs. We obtained designs that came close to ideal, but, consistent with the above cited literature, we were unable to hand tune these mechanisms to fully satisfy design goals. E.g., for the jump-over goal we found a design that often cleared the wall but did not land upright; see Figure 4-24, left. These mechanisms are our *initial* designs.

Presuming our initial designs are potentially close to successful designs, we perform local optimization over a pair of key design parameters, here generally length and height (see Figure 4-18), to seek a nearby solution. For each mechanism we pose its design goal as an objective and a set of constraints; these functions are detailed in Figure 4-17. To evaluate these objectives and constraints we apply DAC and BBI simulation at each design sample queried by the optimizer. Statistics for the numbers of simulation samples, iterations and timings for performing these local optimizations are summarized in Figure 4-17.

For all these examples we found that a relatively small number of iterations were needed to find successful designs; this suggests that our initial, hand-tuned designs are close to solutions in a basin. However, practical design optimization would demand a global optimization strategy to resolve non-convexity. In such general cases initial designs can be expected to start far from optima while the search space is often large - local optimization would be insufficient.

The mechanisms found by this process are our *final* designs. We compare the trajectories and outcomes predicted by BBI and DAC against the real world initial and final mechanisms: validation results are detailed for each example below in Section 4.5.4, while our user study results are presented in Section 4.5.5. The uncut video footage of all experiments are available online Chen et al. [2017].

Design Task	Objectives	Constraints	Iterations	Simulations	Time(m)
Flipper	θ_c^2	—	5	44	242
Catapult	$\|\|x_c - x_{target}\|\|$	—	5	39	350
Jump Onto	θ_c^2	$x_{wall}-x_c \leq 0$, $y_{wall}-y_c \leq 0$	5	47	376
Jump Over	θ_c^2	$x_{wall}-x_c \leq 0$, $y_{wall}-y_{highest} \leq 0$	7	56	415

Figure 4-17: Design optimization statistics. For each dynamic design optimization we report the design task objectives and constraints applied, the number of optimization iterations performed, the total number of simulations performed for each design and finally the total wall-clock time (minutes) spent in design optimization. Sampled frames for each simulated and corresponding fabricated mechanism designs are given in Figures 4-21–4-25 and design parameters optimized over are summarized in Figure 4-18. Here θ_c and $\mathbf{x}_c$ represent jumper angle of rotation and catapult projectile center-of-mass position at landing while $\mathbf{x}_{\text{target}}$ is the desired projectile target.

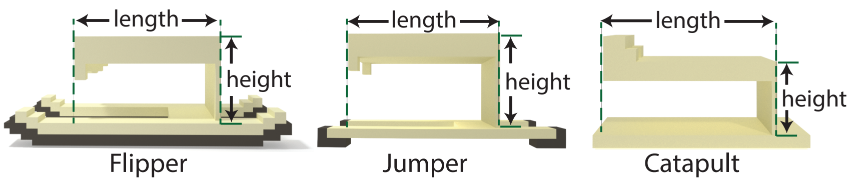


Figure 4-18: Design geometries and parameters.

4.5.2 Implementation

All results are computed on an Amazon EC2 compute-optimized instance with 4 CPU threads (Intel Xeon E5-2666 v3, 2.9Ghz), while all mechanisms were printed on a Objet500. While DAC and BBI give us orders of magnitude speedups for predictive simulation of deforming dynamics, our experiments (see Supplemental) show that dynamic FE simulation is unnecessary when modeling initial loading as well as during some portions of free-flight. With careful book-keeping and mapping of state simpler and more efficient models can be employed during these phases to gain further speedup. When initially loading mechanisms, e.g., when pressing down a jumper in Figure 4-5, we observe that the process is effectively quasistatic and so simulate with an efficient quasistatic solver detailed in our Supplemental. Upon completion of the initial loading we map state to our full dynamic solver with DAC and BBI. We also track the time-varying elastic potential energy stored in simulated compliant objects. When damping causes this internal potential to fall to zero, e.g., during portions of free-flight, we switch from our full DAC discretization to a rigid body discretization. We use DMV [Moser and Veselov, 1991], an efficient energy–momentum preserving rigid-body integrator, to then time step the system in SE(3) coordinates until the next collision is reached, at which point we map rigid-body state back to the DAC model to capture the new deformation dynamics at impact. See our Supplemental for details on this process.

4.5.3 Identification

	Young's Modulus (GPa)			
Material (model)	Reported	Identified	Numerical	Damping
Rigur, vertical (jumper)	1.9 ± 0.2	1.66	0.65	1.64E-04
Rigur, horizontal (jumper)	1.9 ± 0.2	2.06	0.81	1.40E-04
DM4825, horizontal (plant)	1.2 ± 0.3	1.57	0.72	2.55E-04
TangoBlackPlus (jumper)	(2±1)E-04	6.32E-04	4.625E-04	9.10E-02

Figure 4-19: Material parameters identified by our fine and coarse matching against calibration. Left to right we list previously reported Young's moduli for vertically and horizontally oriented prints compared with our identified moduli. We then report the matched numerical moduli we use for each for our DAC models and finally, list previously unreported damping parameters we identify and use in our simulations.

We compute each DAC model's coarse mesh resolution and material parameters using the measurement procedure described above in Section 4.3; see also Figure 4-20. We model heterogeneous materials by computing numerical Young's moduli for soft (TangoBlackPlus) and rigid (Rigur RGD450) materials and coarsening as much as possible while retaining a single material per element. A major benefit of our coarsening scheme is that it identifies both actual and numerical moduli and damping parameters of real-world materials. Figure 4-19 details these parameters. The damping parameters we present here have not, to our knowledge, been previously identified in the literature. On the other hand, in our experiments and simulations we find that the previously identified Poisson's ratio for our printed

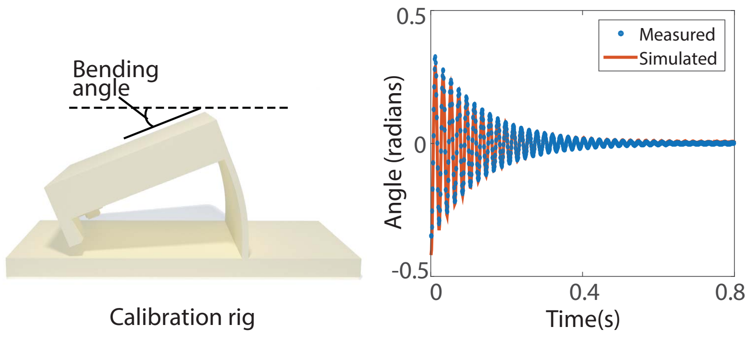


Figure 4-20: Measured and simulated vibrations of a 3D-printed jumper (Rigur) model (magnify the plot for details of fit).

materials Major et al. [2011], at 0.45, is effective, and keep it fixed at this value for all examples reported here.

4.5.4 Experiments

In this section we detail the individual design examples. We visually compare the trajectory behavior of both the real world fabricated initial and final mechanisms with corresponding DAC and BBI simulation results at the same parameters. In the following section we then discuss the results of the user study we perform to evaluate the results of initial and final fabricated mechanisms over repeated user trials comparing against DAC and BBI predicted simulation outcomes.

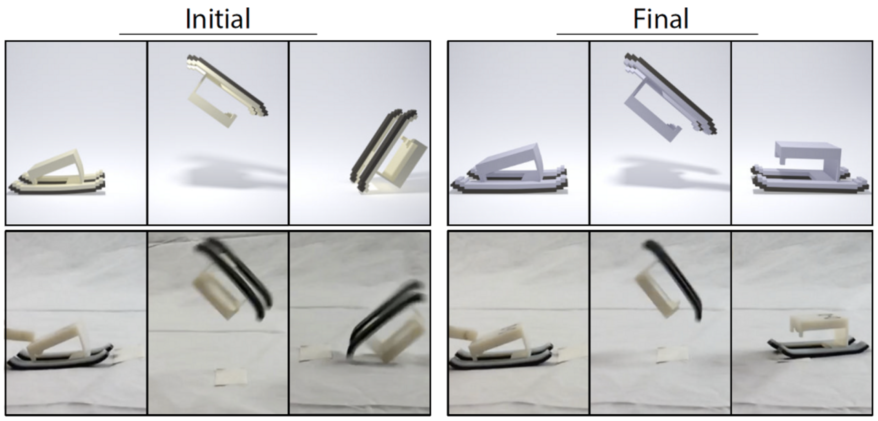


Figure 4-21: A comparison of experimental (bottom) and DAC/BBI simulated (top) results for initial (left) and final (right) designs of a flipping mechanism.

Flipper Our first design example is a simple forward flipping jumper. We begin with the geometry in Figure 4-21 left, load the jumping model by pressing down and then releasing. The design goal is a shape that, upon release, jumps forward into the air, flips and then lands stably on its feet, see e.g., Figure 4-5. See Figure 4-21 comparing simulated and experimental trajectories for both initial and final design samples.

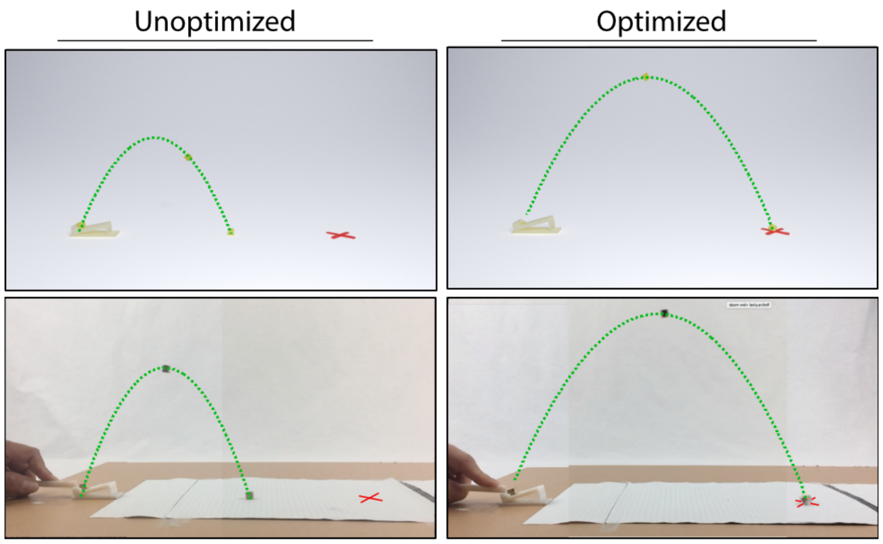


Figure 4-22: A comparison of experimental (bottom) and DAC/BBI simulated (top) results for initial (left) and final (right) designs of the catapult mechanism.

Catapult Our next design example is a catapult mechanism. By adding a firing basket to the above flipper geometry, fixing the mechanism base to the ground, and then adding a projectile cube in the basket to the design model we obtain a catapult for throwing metal cubes at targets. This system requires modeling the sliding contact and impact between the cube and basket. The design goal here is to find a catapult geometry that, under loading produces the correct combination of launch position and release velocity for the catapult arm and block so that the block hits a predetermined target. See Figure 4-22 comparing simulated and experimental trajectories for both initial and final design samples.

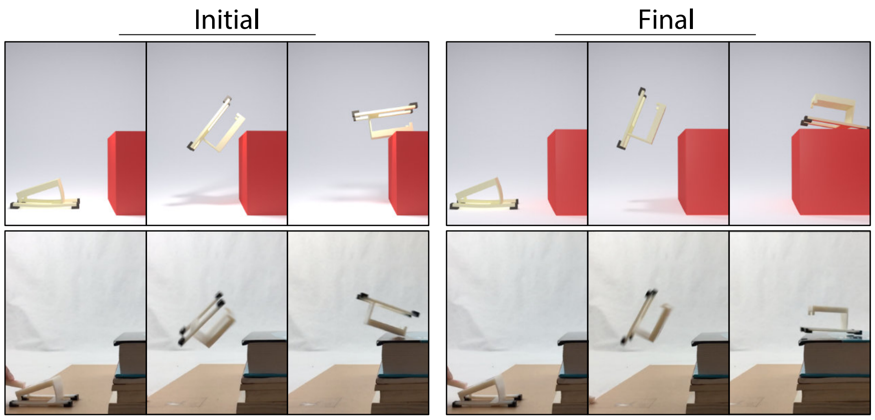


Figure 4-23: A comparison of experimental (bottom) and DAC/BBI simulated (top) results for initial (left) and final (right) designs for a jumper mechanism to jump onto a platform and land upright.

Jumping onto obstacles Here we consider a design example where the goal is to find a geometry for a 3D-printed jumper mechanism that, upon release, jumps forward and upwards into the air, flips (possibly multiple times) and then lands stably upon its feet on top of a flat obstacle, see e.g., Figure 4-23. See Figure 4-23 comparing simulated and experimental trajectories for both initial and final design samples.

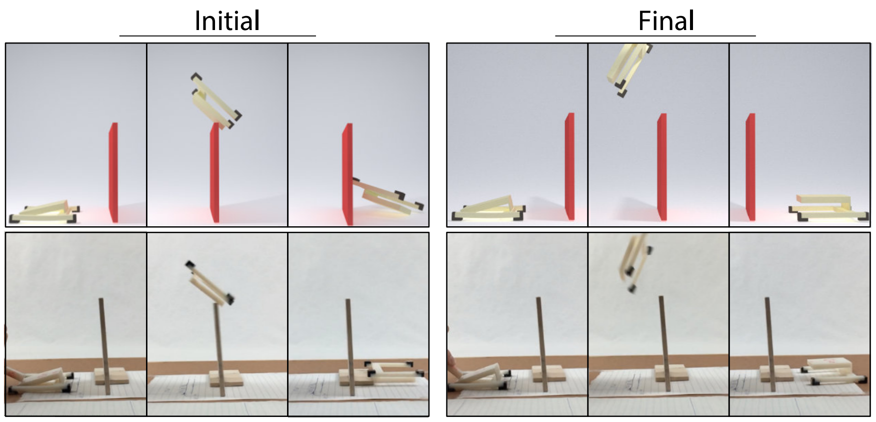


Figure 4-24: A comparison of experimental (bottom) and DAC/BBI simulated (top) results for initial (left) and final (right) designs for a jumper mechanism to jump over a wall of specified height and land upright.

Jumping over obstacles In this example, the goal is to find a geometry for a 3D-printed mechanism that, upon release, jumps forward and upwards high enough into the air to clear a wall, flip (possibly multiple times) over it and then land stably on the other side. See Figure 4-24 comparing simulated and experimental trajectories for both initial and final design samples.

Flipper variations We additionally evaluate six further design example comparisons between fabricated mechanisms and DAC and BBI simulated trajectories over a range of flipper mechanism variations. In Figure 4-25 we compare initial and final designs created by three variations away form the base initial ("normal") flipper mechanism design.

4.5.5 User Study Results

Here we present the results of a user study evaluating the goal outcomes of initial and final fabricated mechanisms over repeated user trials as compared against outcomes predicted by DAC and BBI.

We asked five users to perform twenty attempts each with both the initial and final versions of each of the above mechanisms; the flipper, catapult, jump onto and jump over. For the flipper, jump onto and jump over mechanisms we count the number of successful attempts for each user; see Figure 4-27. Here we define goal success as satisfying the objectives and constraints posed by each design as described above; e.g. flipping, clearing all obstacles, and landing feet down on the desired area (Figures 4-21, 4-23 and 4-24). In Figure 4-27 we summarize the total number of successes for each user per optimized and unoptimized design as well as the aggregate totals. For the optimized and unoptimized catapult designs

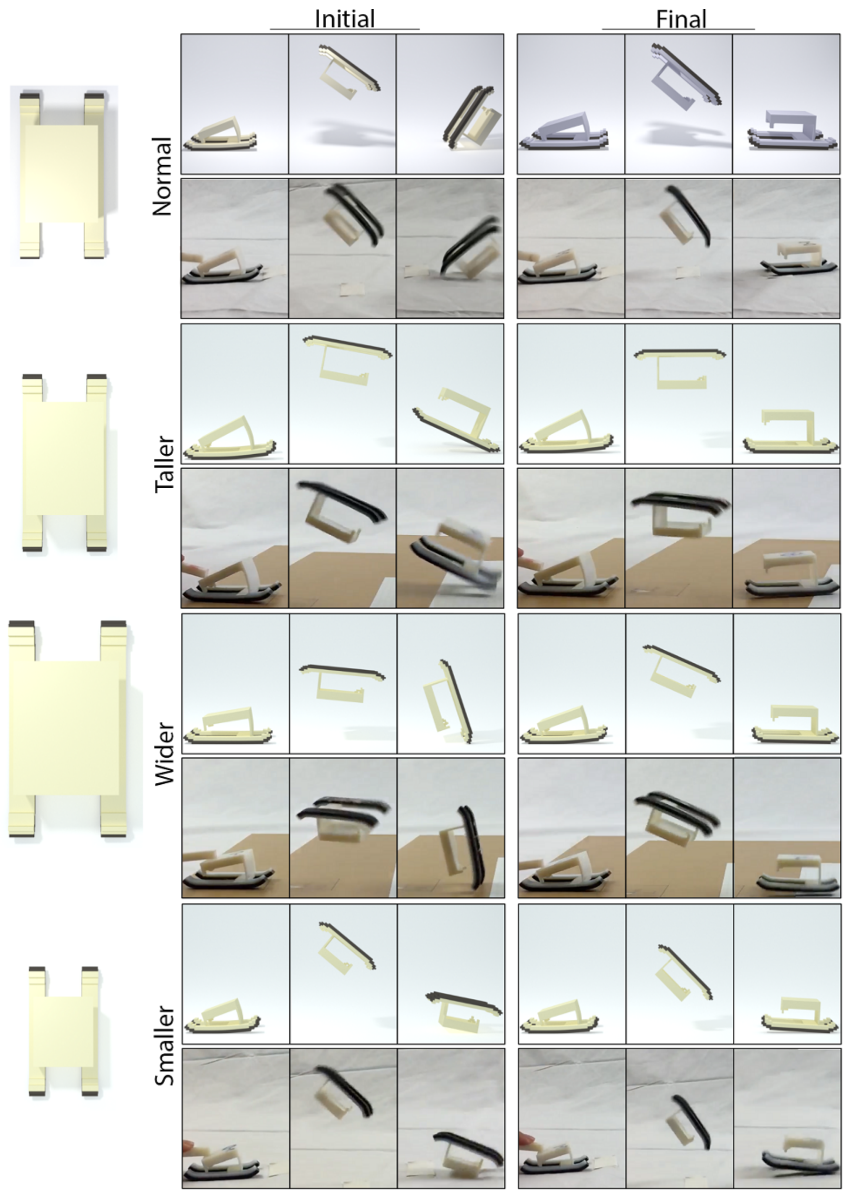


Figure 4-25: A comparison of experimental (bottom) and DAC/BBI simulated (top) results for six more design samples: initial (left) and final (right) designs of variations on the flipping mechanism ("Normal").

we report the mean distance to the target in millimeters and the standard deviation for each user (Figure 4-26).

In order to consistently apply the loading force to each mechanism, users are instructed to fully load each mechanisms by pressing the top until it makes contact with the bottom. We ask users to apply the loading force with a 3D printed bar at a designated loading point marked on each mechanism with permanent marker. The user then deformed the mechanism to the loaded state and released the load with a sliding motion, similar to the launching motion in Tiddlywinks. This ensures that contact break between the stick and the mechanism is close-to instantaneous and consistent. The same criteria are used in simulation to obtain

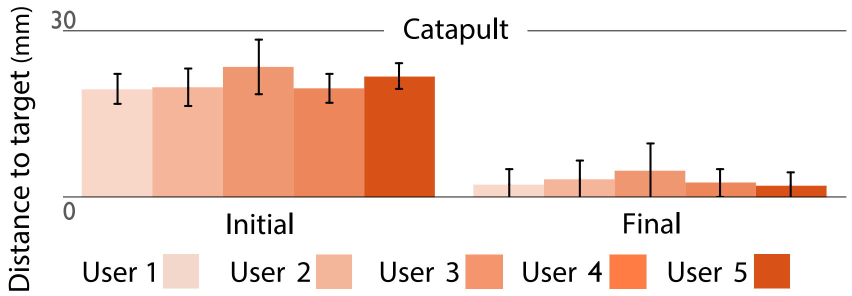


Figure 4-26: Statistics summarizing our user study comparing results from trials with our initial and final *catapult* mechanisms. We report the distance to target per trial. The final catapult design dramatically outperforms its initial counterpart, consistently, across all users, coming close to matching DAC/BBI predicted outcomes.

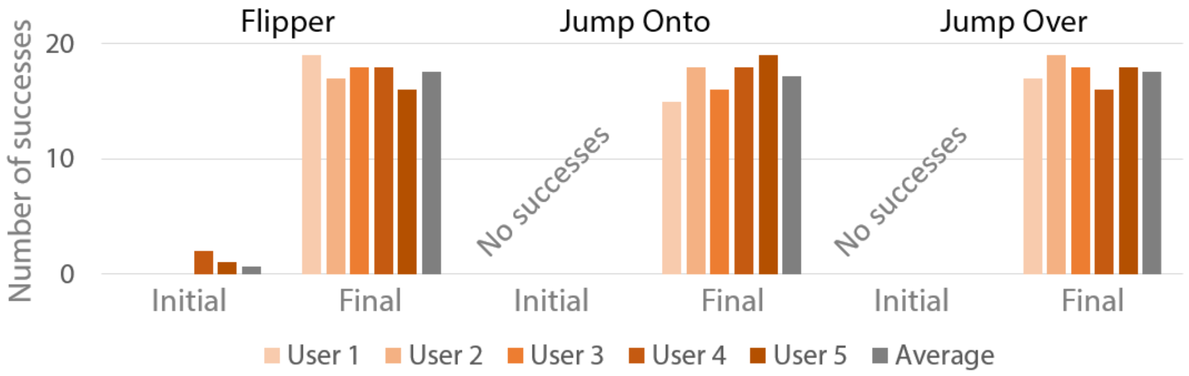


Figure 4-27: Statistics summarizing our user studies comparing results from trials (left to right) with our our initial and final *flipper*, *jump-onto*, and *jump-over* mechanisms. In orange bars we report the number of successes per user, while in grey we report total successes per mechanism across all trials. Note that initial designs for both the *jump-onto* and *jump-over* have no successes at all, while, across all three mechanisms, final designs dramatically outperform their initial counterparts. This success is consistent across all users - closely matching DAC/BBI predicted outcomes.

comparable loading. The Objet500 we employ in all our fabrication examples works at 85 microns precision while our design parameter changes are on the order of millimeters and are thus be reliably manufactured. Additionally to further minimize variability in experimental conditions we use the same printer for all examples and always print in the top left area of the build tray. We orient models consistently and perform experiments within two weeks of printing to avoid long-term material degradation - an interesting topic for future modeling and design research.

Throughout the study, mechanism goal outcomes consistently matched those predicted by their corresponding DAC/BBI simulations. Final, optimized, mechanisms were much more reliable than their initial counterparts - under simulation these were successful. In terms of jumping (flipping, over, onto) tasks final designs completed more than 85% of their attempts for every task - close to matching DAC/BBI predicted success. while the initial designs successfully completed the simplest flipping task in 3% of all attempts (Figure 4-27) and had zero successes for the jump-onto and jump-over tasks - matching DAC/BBI predicted failure. This large difference validates that designed mechanism successes are quite reliably modeled by DAC/BBI. For instance, our catapult design achieves a 10× reduction in mean error with respect to target distance for all users. Note that here there is a small increase in standard deviation due to the fact that our optimized design of the catapult necessarily throws the cube much further, amplifying any variation errors in initial targeting (Figure 4-26). For supporting evidence of each mechanism's experimental behavior please see our supplemental materials which include uncut videos Chen et al. [2017] of all user studies.

Chapter 5

Conclusion

We presented a class of multiscale methods for efficient FEM simulation of elastic materials. Our goal is to use simulation and computational design algorithms to improve real-world designs. To this end, we developed simulation tools for both linear and non-linear elastic materials. DDFEM simulates non-linear statics with two-orders of magnitude speed up while capturing the macroscopic behavior much more accurately than the baseline methods. DAC achieved a 79 times speed up of dynamic simulation while matching the trajectories of simulation and physical measurement. Our simulation algorithms are efficient enough to be used in iterative design algorithms and also predictive enough to improve real-world designs. The accuracy and efficiency of our methods have been demonstrated by designing and fabricating functional 3D prints that meet specified static and dynamic deformation properties. Our two-scale topology optimization algorithm have been shown to optimize designs with 1 trillion voxel on a single computer. DAC is the first algorithm to optimize physical elastic dynamic mechanisms that undergo loading, frictional contact and high-speed impact.

Traditionally, engineers improve the accuracy of FEM simulation by adaptively refining the mesh until some convergence test is satisfied. The elements must be small enough to capture the geometric and material variations. Moreover, to combat numerical stiffening, the element sizes must be a fraction of the geometric feature sizes. Because many small elements are needed to represent a detailed design, accurate simulation can take from minutes to hours on a single computer. Iterative design processes often requires tens or hundreds of simulations, making it impractical to use accurate simulation in the design loop. Recently, researchers tried to take a different route and proposed to correct the material models instead of refining the coarse elements. Homogenization and numerical coarsening methods are examples of this strategy where they use different rules to compute new material models for the coarse elements. Our coarsening approach generalizes the previous methods to handle non-linear elastic materials and dynamic simulations. The energy function of a coarse element is compactly represented as a weighted sum of basis terms. Our simulation relies on a precomputation stage that constructs a database of material combinations and their corresponding coarse energy function parameters. At runtime, our algorithm coarsens the high resolution mesh and then quickly finds the proper material parameters for each coarse element. Coarsening reduces the problem size to only a fraction of the original size. The reduction in problem size leads to dramatic speed improvements. To account for the numerical error caused by insufficient number of degrees of freedom, we simulate the coarse elements using precomputed material properties.

To apply our simulation to computational design algorithms, we developed a set of tools for the intermediate steps. The level set representation of the material property gamut is one such example. It enables efficient sampling and expansion of the material property gamut. With the gamut of 3D cubic-symmetric microstructures, we performed more analysis to identify similarities between structures with extremal properties. This study lead to the discovery of 5 families of auxetic microstructures. The level set gamut has also been used in our two-scale topology optimization algorithm to constrain the material parameters at each cell. Our algorithm can optimize much more complex models since each cell now contains material parameters that can be mapped to precomputed microstructures. Another tool developed in the work is the boundary-balancing impact (BBI) model. It corrects the overly energetic rebound behavior of elastic objects undergoing inelastic impact. This makes the simulation algorithm to be predictive enough to produce the same qualitative landing behavior for the jumping mechanisms.

Our simulation algorithms are used by computational design tools to generate elastic objects satisfying design goals such as using the minimum amount of material and achieving target deformed shapes. In addition to software validation, our simulation is extensively validated by physical experiments using fabricated objects. For static objects, we validated their Young's modulus and Poisson's ratio using compression tests. The fabricated compliant mechanisms also deform to target shapes. For dynamic mechanisms, our simulation replicates the behavior of hand-tuned and optimized jumping mechanisms. The landing poses and rebound are accurately predicted. The average behavior across many trials by different users also agrees with our simulation prediction. Our experiments points the way forward for our simulations to be extended and tested for more complex deformable and dynamic systems.

5.1 Limitations and Future Work

Our work is a first step towards efficient computational design of physical objects. There are many potential future directions for improvements, experiments and applications.

Generalize material models Our material model is similar to traditional material models. It works with deformation gradients sampled at quadrature points. However, in general, the elastic energy can be any function of the vertex positions that respects conservation laws. One can use a much broader class of functions such as a neural network that map from a vector of vertex displacements to an energy value. A more general function can capture more accurately the behavior of higher order deformations such as bending and twisting.

Design multiple physical phenomena Our experiments focused on elastic properties. Engineering design problems cover a much wider range of physical phenomena: structures with zero thermal expansion for space applications, efficient antenna designs from simulation of electric-magnetic field, tougher composite materials by mixing soft and stiff materials and so on. Using similar coarsening techniques to speed up simulation and design of other physical properties remains a promising direction for exploration.

Incorporate other numerical Techniques Our coarsening method and other numerical methods for speed improvements are not mutually exclusive. Since our coarsened discretization uses the same hexahedron elements as the high-resolution elements, we can combine our method with many types of techniques. For example, our current implementation uses a GPU geometric multigrid method to greatly improve the simulation speed compared to a direct linear solver. For future work, we can also experiment with adaptive meshing to use even coarser elements at locations where less details are required. On the other hand, we can learn new material models for higher order elements when more details are required such as with bending and buckling.

Handle high frequency vibration Our work focused on modeling the macroscopic behavior of objects such as the overall deformation and trajectories. Our assumption is that only the low frequency vibrations have sufficient amplitude to influence the macroscopic trajectories. High frequency vibration such as sound are unlikely to be captured accurately out of the box with our current method. To improve the simulation accuracy of such effects, one can embed additional material information such as sound radiation models [Schweickart et al., 2017] in the coarse elements.

Apply non-convex optimization For proof of concept, we only experimented with well-tested optimization algorithms such as topology optimization. For dynamic problems, we applied gradient-based method to improve the design towards a local minimum. We envision future applications such as robot design to require much more sophisticated optimization algorithms that samples and tweaks many types of design parameters such as control parameters, trajectory planning, geometry and material parameters etc. These problems often have many bad local minima, requiring a systemetic sampling of different basins of attraction. Such optimization algorithms would benefit even more from an efficient simulation due to a much larger design space to explore.

Appendix A

Code for Using Reduced Templates

Here we include the Matlab code for generating microstructures given templates and reduced parameters defined in Section 3.3.

```
%\param rparam row vector of 2 reduced parameters in the range
    [-1,1].
%\param family integer family index
%\param res integer of 3D grid resolution
%e.g. st = struct_template([0.1 0.2], 1, 64);
function st = struct_template(rparam, family, res)
  st = zeros(res, res, res);
  %full parameter
  %assume a param.txt included in the same directory.
  [params, withCap]= loadParams('param.txt');
  startIdx = (family-1)*5;
  param = params{startIdx + 1};
  for j = 1:size(rparam,2)
    r = rparam(j);
    if(r>0)
        param = (1-r)*param + r*params{startIdx+2*j};
    else
        param = (1+r)*param - r*params{startIdx+2*j+1};
    end
  end
  nParam = size(param,2);
  paramPerBeam=9;
  nBeam = nParam/paramPerBeam;
  for i = 1:nBeam
      beamParam = [param(1+6*(i-1) : 6*i) param(nBeam*6+1+(i-1)*3
          : nBeam*6+i*3)];
      st = drawCuboid(st, beamParam, withCap(family));
  end
  %enforce cubic symmetry
  %can uncomment to preview a mirrored structure.
  %st = mirrorCubicStructure(st);
end

%\param param parameters for a single cuboid. 6 coordinates
   followed
%cross-section size and orientation.
function o = drawCuboid(arr, param, withCap)
  v0 = param(1:3)';
  v1 = param(4:6)';
  r0= param(7);
  r1 = param(8);
  theta = param(9);
  x = v1-v0;
  len = norm(x);
  o = arr;
  if (len < 1e-10)
    return;
  end
  gridSize = size(arr);
  x = (1.0 / len) * x;
  %default y axis.
  y=[0,1,0]';
  if ( abs(x(2)) > 0.9 )
    y = [1 0 0]';
  end
  z = cross(x, y);
  z=(1/norm(z))*z;
  y = cross(z,x);
  R = vrrotvec2mat([x;theta]);
  y = R*y;
  z = R*z;

  r = sqrt(r0^2 + r1^2);
  r_avg = 0.5 * (r0 + r1);
  r_min = min( r0+r1 * 0.05, r0 * 0.05 + r1);

  %bounding box of the cuboid.
  fl=zeros(3,1);
  fu=zeros(3,1);

  il=zeros(3,1);
  iu=zeros(3,1);
  for i = 1:3
    fl(i) = min(v0(i), v1(i)) - r;
    fl(i) = max(0.0, fl(i));
```

```
      il(i) = floor(fl(i) * gridSize(i) + 0.5);
      fu(i) = max(v0(i), v1(i)) + r;
      fu(i) = min(1.0, fu(i));
      iu(i) = floor(fu(i) * gridSize(i) + 0.5);
    end

    for i = il(1): iu(1)+1
      for j = il(2): iu(2)+1
          for k = il(3):iu(3)+1
          if ~inbound(i, j, k, gridSize)
            continue;
          end
          coord=[(i+0.5)/gridSize(1); (j+0.5)/gridSize(2); (k+0.5)/
             gridSize(3)];
          t = 0;
          [dist, t]= ptLineDist(coord, v0, v1);
          disp = coord - v0;
          ycoord = abs(dot(disp,y));
          zcoord = abs(dot(disp,z));
          if ( ((t >= 0 && t <= 1) || (dist <= r_min && withCap) )&&
              ycoord <= r0 && zcoord <= r1)
            %1-based index for matlab
            arr(i+1,j+1,k+1) = 1;
          end
          end
      end
    end
    o = arr;
  end

  %check if zero-based index within gridSize
  function ret = inbound(i,j,k,s)
   ret = (i>=0 && i<s(1) && j>=0 && j<s(2) && k>=0 && k<s(3));
  end

  function [dist,t] = ptLineDist(pt, x0, x1)
    v0 = pt - x0;
    dir = (x1 - x0);
    len = norm(dir);
    dir = (1.0 / len)*dir;
    %component of v0 parallel to the line segment.
    t = v0'*dir;
    a = t * dir;
    t = t / len;
    b = v0 - a;
    if (t < 0)
      dist = norm(v0);
    elseif (t > 1)
      dist = norm(pt - x1);
    else
      dist = norm(b);
    end
  end

  %\return param cell array of row vectors of parameters.
  function [param, has_cap] = loadParams(filename)
    IN = fopen(filename);
    nRows = fscanf(IN,'%d',1);
    param = cell(nRows,1);
    nTemplate = nRows/5;
    has_cap=fscanf(IN,'%d',nTemplate);
    for i = 1:nRows
        nCol = fscanf(IN,'%d',1);
        arr = fscanf(IN,'%f',[1 nCol]);
        param{i} = arr;
    end
    fclose(IN);
  end
```

The following function mirrors a microstructure to conform to cubic symmetry.

```
function st = mirrorCubicStructure(st_in)
  st = st_in;
  gridSize = size(st_in);
  for i = 0:gridSize(1)-1
    for j = 0:gridSize(2)-1
      for k = 0:gridSize(3)-1
        si = i;
        sj = j;
        sk = k;
        tmp=0;
        if si >= floor(gridSize(1) / 2)
          si = gridSize(1) - i - 1;
        end
        if sj >= floor(gridSize(2)/2)
          sj = gridSize(2) - j - 1;
        end
        if (sk >= floor(gridSize(3)/2))
          sk = gridSize(3) - k - 1;
        end
        if (si < sj)
```

```
          tmp = si;
          si = sj;
          sj = tmp;
        end
        if (si < sk)
          tmp = sk;
          sk = si;
          si = tmp;
        end
        if (sj < sk)
          tmp = sk;
          sk = sj;
          sj = tmp;
        end
        st(i+1,j+1,k+1) = st_in(si+1,sj+1,sk+1);
      end
    end
  end
end
```

We also include an example input file containing template parameters for Template 5. The complete microstructure database, templates and reduced parameters are available online [Chen, 2017].

```
5
0
54
0.2 0.24 0.22 0.52 0.47 0.23
0.2 0.24 0.25 0.21 0.24 0.09
0.45 0.08 0.08 0.28 0.18 0.13
0.52 0.23 0.14 0.26 0.2 0.12
0.43 0.2 0 0.44 0.18 0.16
0.42 0.05 0.06 -0.02 0.01 0.01
0.07 0.05 0.02
0.05 0.02 -1.13
0.08 0.03 0.745
0.06 0.01 0.89
0.07 0.01 -0.08
0.03 0.02 -0.01
54
0.207551 0.202924 0.220949 0.565179 0.473398 0.229526
0.209012 0.240474 0.250949 0.219012 0.24 0.09
0.459051 0.08 0.0899612 0.27 0.179051 0.122411
0.519526 0.220474 0.130988 0.27 0.200474 0.12
0.43 0.199526 -0.00802498 0.439051 0.18 0.150988
0.410513 0.05 0.08 0 0.04 -0.0289737
```

```
0.0690513 0.0495257 0.0205132
0.05 0.02 -1.13
0.0804743 0.03 0.744051
0.058577 0.01 0.887628
0.0695257 0.01 -0.0795257
0.03 0.0290125 0.000513167
54
0.184971 0.230649 0.231241 0.51687 0.506269 0.22438
0.200296 0.24562 0.261241 0.210296 0.24 0.09
0.448759 0.08 0.0915364 0.27 0.168759 0.146565
0.51438 0.22562 0.139704 0.27 0.20562 0.12
0.43 0.19438 0.00940839 0.428759 0.18 0.159704
0.414084 0.05 0.08 0 0.04 -0.0218322
0.0587594 0.0443797 0.0240839
0.05 0.02 -1.13
0.0856203 0.03 0.733759
0.0431392 0.01 0.861899
0.0643797 0.01 -0.0743797
0.03 0.0202958 0.00408392
54
0.2 0.24 0.22 0.52 0.47 0.23
0.2 0.24 0.25 0.21 0.24 0.08
0.46 0.08 0.08 0.29 0.18 0.13
0.52 0.23 0.14 0.26 0.2 0.12
0.43 0.19 0.01 0.44 0.18 0.16
0.43 0.06 0.04 0 0 0
0.07 0.05 0.02
0.05 0.02 -1.13
0.06 0.03 0.745
0.05 0.01 0.89
0.07 0.01 -0.08
0.04 0.02 0
54
0.19 0.22 0.22 0.55 0.49 0.23
0.2 0.24 0.25 0.21 0.24 0.09
0.46 0.08 0.08 0.27 0.18 0.13
0.51 0.22 0.14 0.27 0.2 0.12
0.43 0.2 0 0.44 0.18 0.16
0.42 0.05 0.08 0 0.04 -0.03
0.07 0.05 0.03
0.05 0.02 -1.13
0.08 0.03 0.745
0.06 0.01 0.89
0.07 0.01 -0.08
0.03 0.02 0
```

Appendix B

Implementation and Additional Experiments for Dynamics

B.1 Dynamic Contact-Impact Solver

Implicit time stepping method At each time step our implicit contact method must jointly satisfy the discrete equations of motion

$$\begin{aligned} \mathbf{M}\delta^{t+1} - \mathbf{b}^t - \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^{t+1}) + \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^{t+1})\mathbf{v}^{t+1} \\ - \tfrac{h^2}{2}\mathbf{N}\alpha - \tfrac{h^2}{2}\mathbf{T}\beta = 0, \end{aligned} \tag{B.1}$$

updates

$$\begin{aligned} \mathbf{b}^t &= h\mathbf{M}\mathbf{v}^t + \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^t) - \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^t)\mathbf{v}^t, \\ \mathbf{q}^{t+1} &= \mathbf{q}^t + \delta^{t+1}, \end{aligned} \tag{B.2}$$

contact conditions[1]

$$0 \le \alpha \perp \mathbf{N}^T\delta^{t+1} \ge 0, \tag{B.3}$$

variational maximal dissipation conditions

$$\min_{\beta}\{\beta^T\mathbf{T}^T(\tfrac{2}{h}\delta^{t+1} - \mathbf{v}^t) : \mu_k\bar{\alpha}_k \ge \|\bar{\beta}_k\|,\ \forall k \in \mathcal{C}\}, \tag{B.4}$$

[1]$\mathbf{x} \perp \mathbf{y}$ is the *complementarity condition* $\mathbf{x}_i\mathbf{y}_i = 0,\ \forall i$.

and the impact projection

$$\begin{aligned}\mathbf{c} &= 2\delta^{t+1} - h\mathbf{v}^t,\\ \mathbf{A} &= \mathbf{M} + \frac{h^2}{4}\mathbf{K}(\mathbf{q}^{t+1}) + \frac{h}{2}\mathbf{D}(\mathbf{q}^{t+1}),\\ \mathbf{d}^* &= \operatorname*{argmin}_{\mathbf{d}} \left\{ \|\mathbf{d}-\mathbf{c}\|^2_{\mathbf{A}} : \mathbf{N}^T\mathbf{d} \geq 0 \right\},\\ \mathbf{v}^{t+1} &= \frac{1}{h}\mathbf{d}^*,\end{aligned} \tag{B.5}$$

to convergence with an iterative solver.

Modified Newton-Raphson with frictional contact To construct our solver we first consider time-stepping without contact forces. At each time step we seek a displacement update δ^{t+1} satisfying

$$\mathbf{f}(\delta^{t+1}) = \mathbf{M}\delta^{t+1} - \mathbf{b}^t - \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^{t+1}) + \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^{t+1})\mathbf{v}^{t+1} = 0, \tag{B.6}$$

with

$$\begin{aligned}\mathbf{q}^{t+1} &= \mathbf{q}^t + \delta^{t+1},\\ \mathbf{v}^{t+1} &= \tfrac{2}{h}\delta^{t+1} - \mathbf{v}^t.\end{aligned} \tag{B.7}$$

Ignoring $\frac{\partial \mathbf{D}}{\partial \mathbf{q}}$ we set

$$\mathbf{H}(\delta) = \mathbf{M} + \frac{h^2}{4}\mathbf{K}(\mathbf{q}^t + \delta) + \frac{h}{2}\mathbf{D}(\mathbf{q}^t + \delta). \tag{B.8}$$

and have $\nabla\mathbf{f} \simeq \mathbf{H}$. In the following we will reserve superscripting with indexing t for time step increments and superscripting with indexing i for iteration increments. Modified Newton-Raphson then approximates the linearization of $\mathbf{f}$, at iterate i, around δ^i as

$$\mathbf{f}(\delta^{i+1}) \simeq \mathbf{f}(\delta^i) + \mathbf{H}(\delta^i)(\delta^{i+1} - \delta^i). \tag{B.9}$$

We then find the improved estimate of displacement δ^{i+1} by line search on the descent direction

$$\delta^i - \mathbf{H}(\delta^i)^{-1}\mathbf{f}(\delta^i). \tag{B.10}$$

With contact, at each Newton iterate we are now solving for updated triples of both displacement and boundary contact and friction forces, (δ, α, β). Applying the same modified linearization of (B.1) about δ^{i-1} then gives

$$\mathbf{f}(\delta^{i-1}) + \mathbf{H}(\delta^{i-1})(\delta^i - \delta^{i-1}) - \tfrac{h^2}{2}\mathbf{N}\alpha - \tfrac{h^2}{2}\mathbf{T}\beta = 0, \tag{B.11}$$

Setting

$$\begin{aligned}\mathbf{r}(\delta) = \mathbf{b}^t + \tfrac{h^2}{4}\mathbf{F}(\mathbf{q}^t+\delta) - \tfrac{h^2}{4}\mathbf{D}(\mathbf{q}^t+\delta)(\frac{2}{h}\delta - \mathbf{v}^t) \\ + \big[\frac{h^2}{4}\mathbf{K}(\mathbf{q}^t+\delta) + \frac{h}{2}\mathbf{D}(\mathbf{q}^t+\delta)\big]\delta\end{aligned} \tag{B.12}$$

at each Newton iterate, the linearized contacting system we want to solve is then

$$\begin{aligned}&\mathbf{H}(\delta^{i-1})\delta^i = \mathbf{r}(\delta^{i-1}) + \tfrac{h^2}{2}\mathbf{N}\alpha + \tfrac{h^2}{2}\mathbf{T}\beta,\\ &0 \le \lambda \perp \mathbf{N}^T\delta_{i+1} \ge 0,\\ &\min_\beta\{\beta^T\mathbf{T}^T(\tfrac{2}{h}\delta^{t+1} - \mathbf{v}^t) : \mu_k\bar{\alpha}_k \ge \|\bar{\beta}_k\|,\ \forall k \in \mathcal{C}\}\end{aligned} \tag{B.13}$$

or, equivalently

$$\begin{aligned}0 \le \lambda \perp &\tfrac{h^2}{2}\mathbf{N}^T\mathbf{H}(\delta^{i-1})^{-1}\mathbf{N}\alpha\\ &+ \mathbf{N}^T\mathbf{H}(\delta^{i-1})^{-1}\big[\mathbf{r}(\delta^{i-1}) + \tfrac{h^2}{2}\mathbf{T}\beta\big] \ge 0,\\ \min_\beta\{&\beta^T\mathbf{T}^T(\tfrac{2}{h}\delta^{t+1} - \mathbf{v}^t) : \mu_k\bar{\alpha}_k \ge \|\bar{\beta}_k\|,\ \forall k \in \mathcal{C}\}\end{aligned} \tag{B.14}$$

with the update $\delta^i = \mathbf{H}(\delta^{i-1})^{-1}\big[\mathbf{r}(\delta^{i-1}) + \tfrac{h^2}{2}\mathbf{N}\alpha + \tfrac{h^2}{2}\mathbf{T}\beta\big]$.

Inner-loop solve of Newton steps To solve each Newton step we first backsolve to get

$$\begin{aligned}\tilde{\mathbf{N}} &= \mathbf{H}(\delta^{i-1})^{-1}\mathbf{N},\\ \tilde{\mathbf{D}} &= \mathbf{H}(\delta^{i-1})^{-1}\mathbf{D},\\ \tilde{\mathbf{r}} &= \mathbf{H}(\delta^{i-1})^{-1}\mathbf{r}(\delta^{i-1})\end{aligned} \tag{B.15}$$

The solution then simplifies a bit further to finding

$$\begin{aligned}&0 \le \lambda \perp \mathbf{N}^T\tilde{\mathbf{N}}\alpha + \tilde{\mathbf{N}}^T\big[\tfrac{2}{h^2}\tilde{\mathbf{r}} + \tilde{\mathbf{T}}\beta\big] \ge 0,\\ &\min_\beta\{\beta^T\mathbf{T}^T\tilde{\mathbf{T}}\beta + \beta^T\mathbf{T}^T\big[\tilde{\mathbf{N}}\alpha + \tfrac{2}{h^2}\tilde{\mathbf{r}} - \tfrac{1}{h}\mathbf{v}^t\big]\\ &\qquad : \ \mu_k\bar{\alpha}_k \ge \|\bar{\beta}_k\|,\ \forall k \in \mathcal{C}\}.\end{aligned} \tag{B.16}$$

We then solve the Newton step Gauss-Seidel fashion. Each Gauss-Seidel pass iteratively holds all unknowns except for forces at a single contact $k \in \mathcal{C}$ fixed. We solve for the forces at k, update them and then move on to the next $k+1 \in \mathcal{C}$. We run multiple Gauss-Seidel passes through the system until convergence is reached satisfying (B.16).

To solve for the updated forces $(\bar{\alpha}_k^{i+1}, \bar{\beta}_k^{i+1})$ for contact $k \in \mathcal{C}$ in Gauss-Seidel pass $i+1$

we compute

$$\begin{aligned}\mathbf{d}_k = &\sum_{j<k} \tilde{\mathbf{n}}_j \alpha_j^{i+1} + \sum_{j>k} \tilde{\mathbf{n}}_j \alpha_j^i \\ &+ \sum_{j<k} \tilde{\mathbf{T}}_j \beta_j^{i+1} + \sum_{j>k} \tilde{\mathbf{T}}_j \beta_j^i + \frac{2}{h^2}\tilde{\mathbf{r}}_k \in \mathbb{R}^n.\end{aligned} \tag{B.17}$$

Substituting in (B.16) we then solve the single-point frictional-contact problem at contact k. This is just the small, three-dimensional problem to find $(\bar{\alpha}_k^{i+1}, \bar{\beta}_k^{i+1}) \in \mathbb{R}^3$ satisfying

$$\begin{aligned} 0 \le \bar{\alpha}_k^{i+1} \perp \mathbf{n}_k^T \tilde{\mathbf{n}}_k \bar{\alpha}_k^{i+1} + \mathbf{n}_k^T \tilde{\mathbf{T}}_k \bar{\beta}_k^{i+1} + \mathbf{n}_k^T \mathbf{d}_k \ge 0, \\ \bar{\beta}_k^{i+1} = \underset{\bar{\beta}_k}{\operatorname{argmin}} \{ \bar{\beta}_k \mathbf{T}_k^T \tilde{\mathbf{T}}_k \bar{\beta}_k + \bar{\beta}_k \mathbf{T}_k^T (\tilde{\mathbf{n}}_k \bar{\alpha}_k^{i+1} + \mathbf{d}_k - \tfrac{1}{h}\mathbf{v}^t) \\ : \ \mu_k \alpha_k^{i+1} \ge \|\bar{\beta}_k\| \}. \end{aligned} \tag{B.18}$$

We then update to $(\bar{\alpha}_k^{i+1}, \bar{\beta}_k^{i+1})$ and move on to contact $k+1$.

On convergence of this *inner* Gauss-Seidel iteration to optimal (α^*, β^*) we update to the next Newton step displacement estimate to

$$\delta^i = \tilde{\mathbf{r}} + \tfrac{h^2}{2}\tilde{\mathbf{N}}\lambda^* + \tfrac{h^2}{2}\tilde{\mathbf{T}}\beta^* \tag{B.19}$$

We form the new needed quantities for the next Newton step in (B.15) and then solve the next Newton step with Gauss-Seidel. On convergence of the *outer* Newton iteration to satisfying (B.1), (B.3), and (B.4) we then perform the Impact projection in (B.5) described below and then move to the next time step.

Impact model solve To solve the BBI impact projection step in (B.5) we can reuse the already computed *final* compliant term from the last iterate in (B.15). In dual form our BBI impact projection is equivalent to solving the system

$$0 \le \lambda \perp \mathbf{N}^T \tilde{\mathbf{N}} \lambda + \tilde{\mathbf{N}}^T \mathbf{c} \ge 0, \tag{B.20}$$

and applying a final velocity update

$$\mathbf{v}^{t+1} \leftarrow \frac{1}{h}(\mathbf{c} + \tilde{\mathbf{N}}\lambda), \tag{B.21}$$

with $\mathbf{c}$ given from (B.5). We solve the linear-complementarity problem in (B.20) with projected Gauss-Seidel.

B.2 SE(3) Projections

During free-flight motion we can project each body's FE state to a fitted rigid body model equipped with rotational $\boldsymbol{R} \in SO(3)$ and translational $\boldsymbol{t} \in \mathbb{R}^3$ degrees of freedom. Per body we choose coordinates so that $\boldsymbol{R}$ rotates from principal-axis–aligned body frame to world frame and $\boldsymbol{t}$ gives the location of center of mass. Corresponding angular and linear momenta are $\boldsymbol{\pi}$, $\boldsymbol{l} \in \mathbb{R}^3$, with diagonal inertia tensor I and mass m. Nodal positions of material points $\boldsymbol{x}_i$ during rigid motion are then $\boldsymbol{x}_i^t = \boldsymbol{t}^t + \boldsymbol{R}^t(\boldsymbol{x}_i^0 - \boldsymbol{t}^0)$. We set corresponding momenta to $(\boldsymbol{p}_1^T, ..., \boldsymbol{p}_n^T)^T = \mathbf{M}\mathbf{v}^t \in \mathbb{R}^{3n}$, stack nodal vertices as $\mathbf{Q} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n) \in \mathbb{R}^{3\times n}$ and then project to rigid state with

$$\begin{aligned}
&\boldsymbol{R}^t \leftarrow \underset{\boldsymbol{T}\in SO(3)}{\operatorname{argmin}} ||\boldsymbol{T}\mathbf{Q}^t - \mathbf{Q}^0||_F, \\
&\boldsymbol{\pi}^t \leftarrow \sum_{i=1}^{n} (\boldsymbol{x}_i^t - \boldsymbol{t}^t) \times \boldsymbol{p}_i^t, \quad \boldsymbol{l}^t \leftarrow \sum_{i=1}^{n} \boldsymbol{p}_i^t, \\
&\boldsymbol{I} \leftarrow \int_{\Omega^0} \rho(\boldsymbol{x})[\boldsymbol{x}][\boldsymbol{x}]^T dV, \quad m \leftarrow \int_{\Omega^0} \rho(\boldsymbol{x}) dV.
\end{aligned} \tag{B.22}$$

We then timestep rigid body state forward through free-flight with DMV Moser and Veselov [1991], an energy–momentum preserving rigid-body integrator, until the next collision is reached. Upon collision we again need to model elastic behavior and so project back to closest FE state with nodal positions and velocities

$$\begin{aligned}
&\boldsymbol{x}_i^t \leftarrow \boldsymbol{t}^t + \boldsymbol{R}^t(\boldsymbol{x}_i^0 - \boldsymbol{t}^0), \\
&\boldsymbol{v}_i^t \leftarrow \tfrac{1}{m}\boldsymbol{l}^t - \boldsymbol{R}^t(\boldsymbol{x}_i^0 - \boldsymbol{t}^0) \times (\boldsymbol{I}^{-1}\boldsymbol{\pi}^t).
\end{aligned} \tag{B.23}$$

Comparing the output trajectories between the full FEM simulation and our hybrid projection trajectory we find a tight match throughout. We show the trajectories obtained from full FEM and the hybrid for our *jumping-over* example. Both simulations are initialized to the same configuration and terminate at first impact. Here the jumper traces out an approximately 20cm long trajectory while the two simulated trajectories differ in the L^∞-norm by 0.63 mm for the linear trajectory, i.e., center of mass, and 0.02 radians in rotational pose over the trajectory.

B.3 Static Contact Solver

Loading model We observe in experiment that initial loading processes are effectively quasistatic, with no-slip at contacts. We then model the loading phase with a custom static solver that finds equilibrium state subject to satisfying no-penetration contact constraints $\mathbf{p} \geq 0$ and an assumption of infinite (no-slip) friction. We solve for a δ that gives the constrained equilibrium system maximizing frictional work

$$\begin{aligned}
F(\mathbf{q}^0 + \delta) + \mathbf{N}\alpha + \mathbf{T}\beta + \mathbf{F}_{load} = 0, \\
0 \leq \alpha \perp \mathbf{p}(\mathbf{q}^0 + \delta) \geq 0, \\
\alpha \perp \mathbf{T}^T\delta, \; \forall k \in \mathcal{C}.
\end{aligned} \tag{B.24}$$

Static solver Our static solver applies a direct solution approach to compute nodal positions $\mathbf{q}$ subject to external loads $\mathbf{F}_{ext}$ with infinite-friction (no-slip) unilateral breaking contact. At each iteration step i, we initialize a Newton-Raphson search direction δ^i

$$\mathbf{K}(\mathbf{q}^i)\delta^i = \mathbf{F}_{ext} - \mathbf{F}(\mathbf{q}^i).$$

subject to Dirichlet boundary conditions fixing a set of previously identified *active* contacting boundary vertices identified in the prior iterate. At each iteration we apply a projected line search. We first analyze the depth-component of all the previously identified active contacting points $\mathbf{q}_j$. If $\mathbf{q}_j$ penetrates an obstacle, we half the step size of the search direction until the penetration depth for that point is less than $10^{-2} \times dx$ where dx is characteristic rest element size in our discretization. After applying the line search, we then update the set of active contacting vertices as follows and take the next iterate step. Initially, all vertices touching an obstacle boundary are treated as fixed vertices. We then update the active set of contacting vertices by examining force consistency on all vertices currently contacting the boundary. If the force on a contacting boundary vertex opposes the contacting normal direction, we free it by removing it from the active set; if a contacting boundary vertex was previously free and is now penetrating, we project it back to the contact boundary surface and add it to the active set. We run this solve to convergence satisfying equilibrium in (B.24).

We verify our static solver with respect to full dynamic FEM simulation modeling loading with frictional contact. We find that the relative geometric difference between solutions is 0.2% (Hausdorff distance) while the relative difference between internal energies is 0.4%, with an overall 15X speedup gain from the static solver over the dynamic FEM loading simulation.

B.4 Stiffness Consistent Mass Matrix

We assemble our full mass matrix from element mass matrices. Within each element e, we integrate

$$\mathbf{M}_e = \int_{\Omega_e} \sum_{i=1}^{8} \rho_e \mathbf{N}_i(\xi)\mathbf{N}_i^T(\xi) d\Omega$$

using 2-point Gauss quadrature. Here $\mathbf{N}_i$s are the tri-linear shape functions used for our FE calculations including stress computation. Thus the mass matrix is stiffness-consistent and consistently captures both linear and angular momentum of the system.

B.5 Constraint assembly

For each contact $k \in \mathcal{C}$, the relative acceleration between material points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ (at contact k) can be expressed via the map $\boldsymbol{\Gamma}_k : \dot{\mathbf{q}} \to \dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_j$. If $\boldsymbol{y} \in \mathbb{R}^3$ is a force applied to point $\boldsymbol{x}_i$, and an equal but opposite force is applied to point $\boldsymbol{x}_j$, $\boldsymbol{\Gamma}_k^T \boldsymbol{y}$ is the resulting generalized force applied to the contacting system. For contact k, the map $\boldsymbol{\Gamma}_k$ is the sparse matrix with non-zero entries corresponding to nodes participating in the contacting vertex or vertices. For nodal vertex-boundary contact, a single identity entry corresponding to the node's DoFs is sufficient. For vertex-quadrilateral contacts, we compute the signed identity entries for the five participating face node DoFs weighted by the bilinear weights of the contacting points in the face quadrilateral.

B.6 Experiments on Choosing Predictive Simulation

Accurate physical modeling of transient dynamics with contact is validated by experiment and generally requires simulation at close to convergent spatial and temporal grid sizes. This makes even a single forward dynamic simulation run in 3D prohibitively expensive. On the other end of the spectrum physically based animation methods seek efficiency by pushing simulations towards maximally stable time-step sizes and coarsest possible spatial meshes to obtain visually appealing but generally highly inaccurate dynamics. Here we detail our experiments and investigations towards designing predictive and efficient simulation of high-speed dynamics under frictional contact, impact, loading and free-flight suitable for fabrication design.

Bibliography

Andrew Alderson, Kim L Alderson, Daphne Attard, Kenneth E Evans, Ruben Gatt, Joseph N Grima, W Miller, N Ravirala, CW Smith, and K Zied. Elastic constants of 3-, 4-and 6-connected chiral and anti-chiral honeycombs subject to uniaxial in-plane loading. *Composites Science and Technology*, 70(7):1042–1048, 2010.

G. Allaire and R.V. Kohn. Optimal bounds on the effective behavior of a mixture of two well-ordered elastic materials. *Quaterly of Applied Mathematics*, 1993.

Grégoire Allaire. *Shape optimization by the homogenization method*, volume 146. Springer Science & Business Media, 2012.

Steven S. An, Theodore Kim, and Doug L. James. Optimizing cubature for efficient integration of subspace deformations. *ACM Trans. Graph.*, 27(5):165:1–165:10, December 2008. ISSN 0730-0301.

Ryoichi Ando, Nils Thürey, and Chris Wojtan. Highly adaptive liquid simulations on tetrahedral meshes. *ACM Trans. Graph. (TOG)*, 32(4), 2013.

Erik Andreassen, Boyan S Lazarov, and Ole Sigmund. Design of manufacturable 3d extremal elastic microstructure. *Mechanics of Materials*, 69(1):1–10, 2014.

Sahab Babaee, Jongmin Shim, James C Weaver, Elizabeth R Chen, Nikita Patel, and Katia Bertoldi. 3d soft metamaterials with negative poisson's ratio. *Advanced Materials*, 25(36): 5044–5049, 2013.

Moritz Bächer, Emily Whiting, Bernd Bickel, and Olga Sorkine-Hornung. Spin-it: Optimizing moment of inertia for spinnable objects. *ACM Trans. Graph.*, 33(4):96:1–96:10, July 2014.

Moritz Bächer, Stelian Coros, and Bernhard Thomaszewski. Linkedit: Interactive linkage editing using symbolic kinematics. *ACM Trans. Graph.*, 34(4):99:1–99:8, July 2015.

Jernej Barbič and Doug L. James. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. Graph.*, 24(3):982–990, July 2005.

Jernej Barbič and Yili Zhao. Real-time large-deformation substructuring. *ACM Trans. Graph.*, 30(4):91:1–91:7, jul 2011.

Nicholas W Bartlett, Michael T Tolley, Johannes T B Overvelde, James C Weaver, Bobak Mosadegh, Katia Bertoldi, George M Whitesides, and Robert J Wood. SOFT ROBOTICS. A 3D-printed, functionally graded soft robot powered by combustion. *Science*, 349(6244): 161–165, July 2015.

Ted Belytschko, Wing Kam Liu, Brian Moran, and Khalil Elkhodary. *Nonlinear Finite Elements for Continua and Structures*. John Wiley & Sons, November 2013.

Martin P Bendsøe. Optimal shape design as a material distribution problem. *Structural optimization*, 1(4):193–202, 1989.

Martin P Bendsøe and Ole Sigmund. Material interpolation schemes in topology optimization. *Archive of applied mechanics*, 69(9-10), 1999.

Martin Philip Bendsøe and Ole Sigmund. *Topology Optimization: Theory, Methods, and Applications*. Springer Science & Business Media, 2004.

Sarah Bergbreiter. Effective and efficient locomotion for millimeter-sized microrobots. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4030–4035. IEEE, 2008.

Sarah Bergbreiter and Kristofer SJ Pister. Design of an autonomous jumping microrobot. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 447–453. IEEE, 2007.

Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. Tracks: Toward directable thin shells. *ACM Trans. Graph.*, 26(3):50:1–50:10, July 2007.

Katia Bertoldi, Pedro M Reis, Stephen Willshaw, and Tom Mullin. Negative poisson's ratio behavior induced by an elastic instability. *Advanced Materials*, 22(3):361–366, 2010.

Haimasree Bhatacharya, Yue Gao, and Adam Bargteil. A level-set method for skinning animated particle data. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 17–24. ACM, 2011.

Bernd Bickel, Moritz Bächer, Miguel A. Otaduy, Hyunho Richard Lee, Hanspeter Pfister, Markus Gross, and Wojciech Matusik. Design and fabrication of materials with desired deformation behavior. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 29(4), 2010.

Bernd Bickel, Peter Kaufmann, Mélina Skouras, Bernhard Thomaszewski, Derek Bradley, Thabo Beeler, Phil Jackson, Steve Marschner, Wojciech Matusik, and Markus Gross. Physical face cloning. *ACM Trans. Graph.*, 31(4):118:1–118:10, July 2012.

Jeffrey T Bingham, Jeongseok Lee, Ravi N Haksar, Jun Ueda, and C Karen Liu. Orienting in mid-air through configuration changes to achieve a rolling landing for reducing impact after a fall. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3610–3617. IEEE, 2014.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph.*, 33(4): 154:1–154:11, July 2014.

Tiemo Bückmann, Robert Schittny, Michael Thiel, Muamer Kadic, Graeme W Milton, and Martin Wegener. On three-dimensional dilational elastic metamaterials. *New Journal of Physics*, 16(3):033032, 2014.

Joseph E Cadman, Shiwei Zhou, Yuhang Chen, and Qing Li. On design of multi-functional microstructural materials. *Journal of Materials Science*, 48(1):51–66, 2013.

Vivien J Challis and James K Guest. Level set topology optimization of fluids in stokes flow. *International journal for numerical methods in engineering*, 79(10):1284–1308, 2009.

Desai Chen. *Cubic microstructure database, templates and reduced parameters.*, 2017. `https://www.dropbox.com/sh/oejrntg5murl1bv/AACZv-JxCoRmH5zaekRCz3l7a`.

Desai Chen, David Levin, Wojciech Matusik, and Danny M. Kaufman. *Uncut user study videos*, 2017. `https://www.dropbox.com/sh/5quhghwg6vjjjuz/AAAuBS8ilZ2Tisv52C_YQYrma`.

Xiang Chen, Changxi Zheng, Weiwei Xu, and Kun Zhou. An asymptotic numerical method for inverse elastic shape design. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 33(4), August 2014.

Francisco Chinesta, Adrien Leygue, Felipe Bordeu, Jose Vicente Aguado, Elías Cueto, David González, Iciar Alfaro, Amine Ammar, and Antonio Huerta. Pgd-based computational vademecum for efficient design, optimization and control. *Archives of Computational Methods in Engineering*, 20(1):31–59, 2013.

Kyu-Jin Cho, Je-Sung Koh, Sangwoo Kim, Won-Shik Chu, Yongtaek Hong, and Sung-Hoon Ahn. Review of manufacturing processes for soft biomimetic robots. *International Journal of Precision Engineering and Manufacturing*, 10(3):171–181, 2009.

Wayne A Churaman, Aaron P Gerratt, and Sarah Bergbreiter. First leaps toward jumping microrobots. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, pages 1680–1686, 2011.

Philippe G Ciarlet. *The finite element method for elliptic problems.* SIAM, 2002.

Anders Clausen, Fengwen Wang, Jakob S Jensen, Ole Sigmund, and Jennifer A Lewis. Topology optimized architectures with programmable poisson's ratio over large deformations. *Advanced Materials*, 27(37):5523–5527, 2015.

Pascal Clausen, Martin Wicke, Jonathan R. Shewchuk, and James F. O'brien. Simulating liquids and solid-liquid interactions with Lagrangian meshes. *ACM Trans. Graph.*, 32(2): 17:1–17:15, April 2013.

P.G. Coelho, P.R. Fernandes, J.M. Guedes, and H.C. Rodrigues. A hierarchical model for concurrent material and topology optimisation of three-dimensional structures. *Structural and Multidisciplinary Optimization*, 35(2):107–115, 2008.

Stelian Coros, Bernhard Thomaszewski, Gioacchino Noris, Shinjiro Sueda, Moira Forberg, Robert W. Sumner, Wojciech Matusik, and Bernd Bickel. Computational design of mechanical characters. *ACM Trans. Graph.*, 32(4):83:1–83:12, July 2013.

Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr. Dynamic real-time deformations using space & time adaptive sampling. In *Proc. SIGGRAPH '01*, Annual Conference Series, pages 31–36, 2001.

Peter Deuflhard, Rolf Krause, and Susanne Ertel. A contact-stabilized Newmark method for dynamical contact problems. *International Journal for Numerical Methods in Engineering*, 73(9):1274–1290, 2008.

Yue Dong, Jiaping Wang, Fabio Pellacini, Xin Tong, and Baining Guo. Fabricating spatially-varying subsurface scattering. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 29(4), 2010.

Randal Douc. Comparison of resampling schemes for particle filtering. In *4th International Symposium on Image and Signal Processing and Analysis (ISPA*, pages 64–69, 2005.

David Doyen, Alexandre Ern, and Serge Piperno. Time-Integration Schemes for the Finite Element Dynamic Signorini Problem. *SIAM Journal on Scientific Computing*, 33(1):223–249, January 2011.

Tao Du, Adriana Schulz, Bo Zhu, Bernd Bickel, and Wojciech Matusik. Computational multicopter design. *ACM Trans. Graph.*, 35(6):227:1–227:10, November 2016.

Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Y Zeevi. The farthest point strategy for progressive image sampling. *IEEE Trans. Image Process.*, 6(9):1305–1315, 1997.

Nicholas Fang, Dongjuan Xi, Jianyi Xu, Muralidhar Ambati, Werayut Srituravanich, Cheng Sun, and Xiang Zhang. Ultrasonic metamaterials with negative modulus. *Nature materials*, 5(6):452, 2006.

Charbel Farhat and Francois-Xavier Roux. A method of finite element tearing and interconnecting and its parallel solution algorithm. *INT J NUMER METH ENG*, 32(6):1205–1227, 1991.

CL Farmer. Upscaling: a review. *INT J NUMER METH FL*, 40(1-2):63–78, 2002.

François Faure, Benjamin Gilles, Guillaume Bousquet, and Dinesh K. Pai. Sparse meshless models of complex deformable solids. *ACM Trans. Graph.*, 30(4):73:1–73:10, July 2011.

Lorna J Gibson and Michael F Ashby. The mechanics of three-dimensional cellular materials. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 382, pages 43–59. The Royal Society, 1982.

Benjamin Gilles, Guillaume Bousquet, Francois Faure, and Dinesh K. Pai. Frame-based elastic models. *ACM Trans. Graph.*, 30(2):15:1–15:12, April 2011.

Eitan Grinspun, Petr Krysl, and Peter Schröder. Charms: A simple framework for adaptive simulation. *ACM Trans. Graph.*, 21(3):281–290, July 2002.

José Miranda Guedes and Noboru Kikuchi. Preprocessing and postprocessing for materials based on the homogenization method with adaptive finite element methods. *Comput. Methods Appl. Mech. Eng.*, 83(2):143–198, October 1990.

Chan Soo Ha, Michael E Plesha, and Roderic S Lakes. Chiral three-dimensional isotropic lattices with negative poisson's ratio. *physica status solidi (b)*, 253(7):1243–1251, 2016.

Kris K. Hauser, Chen Shen, and James F. O'Brien. Interactive deformation using modal analysis with constraints. In *Graphics Interface*, pages 247–256. CIPS, Canadian Human-Computer Commnication Society, June 2003.

Miloš Hašan, Martin Fuchs, Wojciech Matusik, Hanspeter Pfister, and Szymon Rusinkiewicz. Physical reproduction of materials with specified subsurface scattering. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 29(4), 2010.

Doug L. James and Dinesh K. Pai. Dyrt: Dynamic response textures for real time deformation simulation with graphics hardware. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 582–585, New York, NY, USA, 2002. ACM. ISBN 1-58113-521-1.

Steven G Johnson. The nlopt nonlinear-optimization package, 2014. URL `http://ab-initio.mit.edu/nlopt`.

Gwang-Pil Jung, Ji-Suk Kim, Je-Sung Koh, Sun-pil Jung, and Kyu-Jin Cho. Role of compliant leg in the flea-inspired jumping mechanism. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, pages 315–320. IEEE, 2014.

Sun-Pill Jung, Gwang-Pil Jung, Je-Sung Koh, Dae-Young Lee, and Kyu-Jin Cho. Fabrication of Composite and Sheet Metal Laminated Bistable Jumping Mechanism. *Journal of Mechanisms and Robotics*, 7(2):021010, February 2015.

Muamer Kadic, Tiemo Bückmann, Nicolas Stenger, Michael Thiel, and Martin Wegener. On the practicability of pentamode mechanical metamaterials. *Applied Physics Letters*, 100 (19):191901, 2012.

Couro Kane, Eduardo A. Repetto, Michael Ortiz, and Jerrold E. Marsden. Finite element analysis of nonsmooth contact. *Computer Methods in Applied Mechanics and Engineering*, 180(1-2):1–26, 1999.

Couro Kane, Jerrold E. Marsden, Michael Ortiz, and Matthew West. Variational integrators and the Newmark algorithm for conservative and dissipative mechanical systems. *International Journal for Numerical Methods in Engineering*, 49(10):1295–1325, 2000.

Lily Kharevych, Patrick Mullen, Houman Owhadi, and Mathieu Desbrun. Numerical coarsening of inhomogeneous elastic materials. *ACM Trans. Graph.*, 28(3):51:1–51:8, July 2009.

Thomas Kiser, Michael Eigensatz, Minh Man Nguyen, Philippe Bompas, and Mark Pauly. Architectural causticsâĂŤcontrolling light with geometry. In *Advances in Architectural Geometry 2012*, pages 91–106. Springer, 2013.

Je-Sung Koh, Sun pil Jung, Robert J Wood, and Kyu-Jin Cho. A jumping robotic insect based on a torque reversal catapult mechanism. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013)*, pages 3796–3801. IEEE, 2013.

Je-Sung Koh, Eunjin Yang, Gwang-Pil Jung, Sun-Pill Jung, Jae Hak Son, Sang-Im Lee, Piotr G Jablonski, Robert J Wood, Ho-Young Kim, and Kyu-Jin Cho. BIOMECHANICS. Jumping on water: Surface tension-dominated jumping of water striders and robotic insects. *Science*, 349(6247):517–521, July 2015.

Xinyu Kou, Geoff T. Parks, and Sooi Thor Tan. Optimal design of functionally graded materials using a procedural model and particle swarm optimization. *Computer-Aided Design*, 44(4):300–310, 2012.

Rolf Krause and Mirjam Walloth. Presentation and comparison of selected algorithms for dynamic contact based on the Newmark scheme. *Applied Numerical Mathematics*, 62(10): 1393–1410, October 2012.

P. Krysl, S. Lall, and J. E. Marsden. Dimensional model reduction in non-linear finite element dynamics of solids and structures. *INT J NUMER METH ENG*, 51(4):479–504, 2001.

Roderic Lakes. Foam structures with a negative poisson's ratio. *Science*, 235:1038–1041, 1987.

Timothy Langlois, Ariel Shamir, Daniel Dror, Wojciech Matusik, and David I. W. Levin. Stochastic structural analysis for context-aware design and fabrication. *ACM Trans. Graph.*, 35(6):226:1–226:13, November 2016.

Ta-Chih Lee, Rangasami L Kashyap, and Chong-Nam Chu. Building skeleton models via 3-d medial surface axis thinning algorithms. *CVGIP: Graphical Models and Image Processing*, 56(6):462–478, 1994.

J Li, L Fok, X Yin, G Bartal, and X Zhang. Experimental demonstration of an acoustic magnifying hyperlens. *Nature materials*, 8(12):931–934, 2009.

Shuguang Li, Robert Katzschmann, and Daniela Rus. A soft cube capable of controllable continuous jumping. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1712–1717. IEEE, 2015.

Siwang Li, Jin Huang, Fernando de Goes, Xiaogang Jin, Hujun Bao, and Mathieu Desbrun. Space-time editing of elastic motion through material optimization and reduction. *ACM Trans. Graph.*, 33(4):108:1–108:10, July 2014. ISSN 0730-0301.

Xiaoyan Li and Huajian Gao. Mechanical metamaterials: Smaller and stronger. *Nature materials*, 15(4):373–374, 2016.

Hod Lipson. Challenges and Opportunities for Design, Simulation, and Fabrication of Soft Robots. *Soft Robotics*, 1(1):21–27, March 2014.

Michael Loepfe, Christoph M Schumacher, Urs B Lustenberger, and Wendelin J Stark. An untethered, jumping roly-poly soft robot driven by combustion. *Soft Robotics*, 2(1):33–41, 2015.

Zoltan Major, Martin Reiter, Elena Hemmeter, and Franz Hiptmair. Combination of novel virtual and real prototyping methods in a rapid product development methodology. *Polimeri*, 32, 2011.

Jan Mandel. Balancing domain decomposition. *COMM NUMER METH ENG*, 9(3):233–241, 1993.

Vladimir A Mandelshtam and Howard S Taylor. Harmonic inversion of time signals and its applications. *The Journal of Chemical Physics*, 107(17):6756–6769, November 1997.

J.E. Marsden and T.J.R. Hughes. *Mathematical Foundations of Elasticity*. Dover Civil and Mechanical Engineering. Dover Publications, 2012.

Jerrold E. Marsden and Matthew West. Discrete mechanics and variational integrators. *Acta Numerica 2001*, 10:357–514, May 2001.

Sebastian Martin, Peter Kaufmann, Mario Botsch, Eitan Grinspun, and Markus Gross. Unified simulation of elastic rods, shells, and solids. *ACM Trans. Graph.*, 29(4):39:1–39:10, July 2010. ISSN 0730-0301.

Tobias Martin, Nobuyuki Umetani, and Bernd Bickel. Omniad: Data-driven omni-directional aerodynamics. *ACM Trans. Graph.*, 34(4):113:1–113:12, July 2015.

Jonàs Martínez, Jérémie Dumas, Sylvain Lefebvre, and Li-Yi Wei. Structure and appearance optimization for controllable shape design. *ACM Trans. Graph.*, 34(6):229:1–229:11, October 2015.

Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph.*, 30(4):37:1–37:12, July 2011.

Geoffrey McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.

Vittorio Megaro, Jonas Zehnder, Moritz Bächer, Stelian Coros, Markus Gross, and Bernhard Thomaszewski. A computational design tool for compliant mechanisms. *ACM Trans. Graph.*, 36(4):82:1–82:12, July 2017.

Lucas R Meza, Satyajit Das, and Julia R Greer. Strong, lightweight, and recoverable three-dimensional ceramic nanolattices. *Science*, 345(6202):1322–1326, 2014.

Graeme W. Milton and Andrej V. Cherkaev. Which elasticity tensors are realizable? *Journal of Engineering Materials and Technology*, 117:483, 1995.

Jürgen Moser and Alexander P Veselov. Discrete versions of some classical integrable systems and factorization of matrix polynomials. *Communications in Mathematical Physics*, 139 (2):217–243, 1991.

Przemyslaw Musialski, Thomas Auzinger, Michael Birsak, Michael Wimmer, and Leif Kobbelt. Reduced-order shape optimization using offset surfaces. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 34(4):102:1–102:9, August 2015.

PB Nakshatrala, DA Tortorelli, and KB Nakshatrala. Nonlinear structural design using multiscale topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 261, 2013.

Rahul Narain, Armin Samii, and James F. O'Brien. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph.*, 31(6):152:1–152:10, November 2012.

Rahul Narain, Tobias Pfaff, and James F. O'Brien. Folding and crumpling adaptive sheets. *ACM Trans. Graph.*, 32(4):51:1–51:8, July 2013.

Matthieu Nesme, Paul G. Kry, Lenka Jeřábková, and François Faure. Preserving topology and elasticity for embedded deformable models. *ACM Trans. Graph.*, 28(3):52:1–52:9, July 2009.

Minkyun Noh, Seung-Won Kim, Sungmin An, Je-Sung Koh, and Kyu-Jin Cho. Flea-Inspired Catapult Mechanism for Miniature Jumping Robots. *IEEE Transactions on Robotics*, 28 (5):1007–1018, 2012.

Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153. Springer Science & Business Media, 2006.

Anna Pandolfi, Couro Kane, Jerrold E. Marsden, and Michael Ortiz. Time-discretized variational formulation of non-smooth frictional contact. *International Journal for Numerical Methods in Engineering*, 53(8):1801–1829, March 2002.

Julian Panetta, Qingnan Zhou, Luigi Malomo, Nico Pietroni, Paolo Cignoni, and Denis Zorin. Elastic textures for additive fabrication. *ACM Trans. Graph.*, 34(4):135:1–135:12, July 2015.

Marios Papas, Wojciech Jarosz, Wenzel Jakob, Szymon Rusinkiewicz, Wojciech Matusik, and Tim Weyrich. Goal-based caustics. *Proc. of Eurographics*, 30(2):503–511, June 2011.

Taylor Patterson, Nathan Mitchell, and Eftychios Sifakis. Simulation of complex nonlinear elastic bodies using lattice deformers. *ACM Trans. Graph.*, 31(6):197:1–197:10, November 2012.

Jaime Peraire, Joaquim Peiró, and Ken Morgan. A 3D finite element multigrid solver for the Euler equations. *AIAA*, 92, 1992.

D Prall and RS Lakes. Properties of a chiral honeycomb with a poisson's ratio ofâĂŤ1. *International Journal of Mechanical Sciences*, 39(3):305–314, 1997.

Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. Make it stand: Balancing shapes for 3d fabrication. *ACM Trans. Graph.*, 32(4):81:1–81:10, July 2013.

Pedro M. Reis. A Perspective on the Revival of Structural (In) Stability With Novel Opportunities for Function: From Buckliphobia to Buckliphilia. *Journal of Applied Mechanics*, 82(11):111001, 2015.

Pedro M. Reis, Heinrich M. Jaeger, and Martin van Hecke. Designer Matter: A perspective. *Extreme Mechanics Letters*, 5:25–29, December 2015.

Daniel Ritchie, Ben Mildenhall, Noah D. Goodman, and Pat Hanrahan. Controlling procedural modeling programs with stochastically-ordered sequential monte carlo. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 34(4), 2015.

H. Rodrigues, Jose M. Guedes, and M.P. Bendsøe. Hierarchical optimization of material and structure. *Structural and Multidisciplinary Optimization*, 24(1):1–10, 2002.

JH Roh, CB Giller, PH Mott, and CM Roland. Failure of classical elasticity in auxetic foams. *AIP Advances*, 3(4):042126, 2013.

Daniela Rus and Michael T. Tolley. Design, fabrication and control of soft robots. *Nature*, 521(7553):467–475, May 2015.

Krishna Kumar Saxena, Raj Das, and Emilio P Calius. Three decades of auxetics research-materials with negative poisson's ratio: a review. *Advanced Engineering Materials*, 18(11): 1847–1870, 2016.

Christian Schumacher, Bernd Bickel, Jan Rys, Steve Marschner, Chiara Daraio, and Markus Gross. Microstructures to control elasticity in 3D printing. *ACM Trans. Graph.*, 34(4), 2015.

David Schurig, JJ Mock, BJ Justice, Steven A Cummer, John B Pendry, AF Starr, and DR Smith. Metamaterial electromagnetic cloak at microwave frequencies. *Science*, 314 (5801):977–980, 2006.

Eston Schweickart, Doug L James, and Steve Marschner. Animating elastic rods with sound. *ACM Transactions on Graphics (TOG)*, 36(4):115, 2017.

Ahmed A. Shabana. *Theory of Vibration, Vol. II.* Springer-Verlag, Berlin, 1991.

Ahmed A Shabana. *Theory of Vibration* . Volume I: An Introduction. Springer Science & Business Media, New York, NY, December 2012.

Vladimir M Shalaev. Optical negative-index metamaterials. *Nature photonics*, 1(1):41–48, 2007.

Ole Sigmund. Materials with prescribed constitutive parameters: an inverse homogenization problem. *International Journal of Solids and Structures*, 31(17):2313–2329, 1994.

Ole Sigmund. On the design of compliant mechanisms using topology optimization. *Mechanics of Structures and Machines*, 25(4), 1997.

Ole Sigmund. A new class of extremal composites. *Journal of the Mechanics and Physics of Solids*, 48(2):397 – 428, 2000.

Ole Sigmund. Morphology-based black and white filters for topology optimization. *Structural and Multidisciplinary Optimization*, 33:401–424, 2007.

Ole Sigmund and Kurt Maute. Topology optimization approaches. *Structural and Multidisciplinary Optimization*, 48(6):1031–1055, 2013.

Mélina Skouras, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, and Markus Gross. Computational design of actuated deformable characters. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 32(4), 2013.

Ondrej Stava, Juraj Vanek, Bedrich Benes, Nathan Carr, and Radomír Měch. Stress relief: Improving structural strength of 3d printable objects. *ACM Trans. Graph.*, 31(4):48:1–48:11, July 2012.

Krister Svanberg. The method of moving asymptotes —a new method for structural optimization. *International journal for numerical methods in engineering*, 24(2):359–373, 1987.

Barna Szabó, Alexander Düster, and Ernst Rank. The p-version of the finite element method. *Encyclopedia of Comput. Mech.*, 2004.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Rosell Torres, Alejandro Rodríguez, José M. Espadero, and Miguel A. Otaduy. High-resolution interaction with corotational coarsening models. *ACM Trans. Graph.*, 35(6): 211:1–211:11, November 2016. ISSN 0730-0301.

Erva Ulu, James Mccann, and Levent Burak Kara. Lightweight structure design under force location uncertainty. *ACM Trans. Graph.*, 36(4):158:1–158:13, July 2017.

Nobuyuki Umetani, Yuki Koyama, Ryan Schmidt, and Takeo Igarashi. Pteromys: Interactive design and optimization of free-formed free-flight model airplanes. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 33(4):65:1–65:10, July 2014.

Dominic Vella. ROBOTICS. Two leaps forward for robot locomotion. *Science*, 349(6247): 472–473, July 2015.

Panagiotis Vogiatzis, Shikui Chen, Xiao Wang, Tiantian Li, and Lifeng Wang. Topology optimization of multi-material negative poissonâĂŹs ratio metamaterials using a reconciled level set method. *Computer-Aided Design*, 83:15–32, 2017.

Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1):25–57, May 2006. ISSN 0025-5610.

Bin Wang, Longhua Wu, KangKang Yin, Uri Ascher, Libin Liu, and Hui Huang. Deformation capture and modeling of soft objects. *ACM Trans. Graph.*, 34(4):94:1–94:12, July 2015. ISSN 0730-0301.

Qiming Wang, Julie A Jackson, Qi Ge, Jonathan B Hopkins, Christopher M Spadaccini, and Nicholas X Fang. Lightweight mechanical metamaterials with tunable negative thermal expansion. *Physical review letters*, 117(17):175901, 2016.

Tim Weyrich, Pieter Peers, Wojciech Matusik, and Szymon Rusinkiewicz. Fabricating microgeometry for custom surface reflectance. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 28 (3):32:1–32:6, July 2009.

Martin Wicke, Daniel Ritchie, Bryan M. Klingner, Sebastian Burke, Jonathan R. Shewchuk, and James F. O'Brien. Dynamic local remeshing for elastoplastic simulation. *ACM Trans. Graph.*, 29(4):49:1–49:11, July 2010.

Jun Wu, Christian Dick, and Rudiger Westermann. A system for high-resolution topology optimization. *IEEE Transactions on Visualization and Computer Graphics*, 22(3):1195–1208, March 2016.

Liang Xia and Piotr Breitkopf. A reduced multiscale model for nonlinear structural topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 280:117–134, 2014.

Liang Xia and Piotr Breitkopf. Design of materials using topology optimization and energy-based homogenization approach in matlab. *Structural and Multidisciplinary Optimization*, pages 1–13, 2015a.

Liang Xia and Piotr Breitkopf. Multiscale structural topology optimization with an approximate constitutive model for local material microstructure. *Computer Methods in Applied Mechanics and Engineering*, 286:147–167, 2015b.

X Yan, X Huang, Y Zha, and YM Xie. Concurrent topology optimization of structures and their composite microstructures. *Computers & Structures*, 133:103–110, 2014.

Xiaolei Yan, Xiaodong Huang, Guangyong Sun, and Yi Min Xie. Two-scale optimal design of structures with thermal insulation materials. *Composite Structures*, 120:358–365, 2015.

Xiaoyu Zheng, Howon Lee, Todd H Weisgraber, Maxim Shusteff, Joshua DeOtte, Eric B Duoss, Joshua D Kuntz, Monika M Biener, Qi Ge, Julie A Jackson, et al. Ultralight, ultrastiff mechanical metamaterials. *Science*, 344(6190):1373–1377, 2014.

Qingnan Zhou, Julian Panetta, and Denis Zorin. Worst-case structural analysis. *ACM Trans. Graph.*, 32(4):137:1–137:12, July 2013.

Bo Zhu, Melina Skouras, Desai Chen, and Wojciech Matusik. Two-scale topology optimization with microstructures. *ACM Trans. Graph.*, 36(4), July 2017.

Lifeng Zhu, Weiwei Xu, John Snyder, Yang Liu, Guoping Wang, and Baining Guo. Motion-guided mechanical toy modeling. *ACM Trans. Graph.*, 31(6):127:1–127:10, November 2012.

Yongning Zhu and Robert Bridson. Animating sand as a fluid. *ACM Trans. Graph.*, 24(3): 965–972, 2005.

Yongning Zhu, Eftychios Sifakis, Joseph Teran, and Achi Brandt. An efficient multigrid method for the simulation of high-resolution elastic solids. *ACM Trans. Graph.*, 29(2): 16:1–16:18, April 2010.